

PROJECTS ■ THEORY ■ APPLICATIONS ■ CIRCUITS ■ TECHNOLOGY

# NUTS AND VOLTS

www.nutsvolts.com  
October 2015

EVERYTHING FOR ELECTRONICS

## Setting Up Your TEST BENCH

BREAKING THE  
ARDUINO  
SPEED LIMIT

*A close-up  
look at the  
essential*

**MUST  
HAVE  
GEAR**



FERROELECTRIC CAPACITORS  
VINTAGE COMPUTING  
UNDERSTANDING HARMONICS

# Introducing The New X Series!



- Large 8 inch color LCD
- 100MHz, 200MHz, bandwidth models
- Real-time sampling rate up to 1GSa/s
- Record length of 14Mpts, Waveform capture rate up to 60,000 wfs/s
- New generation of SPO technology
- Supports 256-level intensity grading and color temperature display
- SPO Technology means low noise

## SDS 1000X Series Super Phosphor Oscilloscope



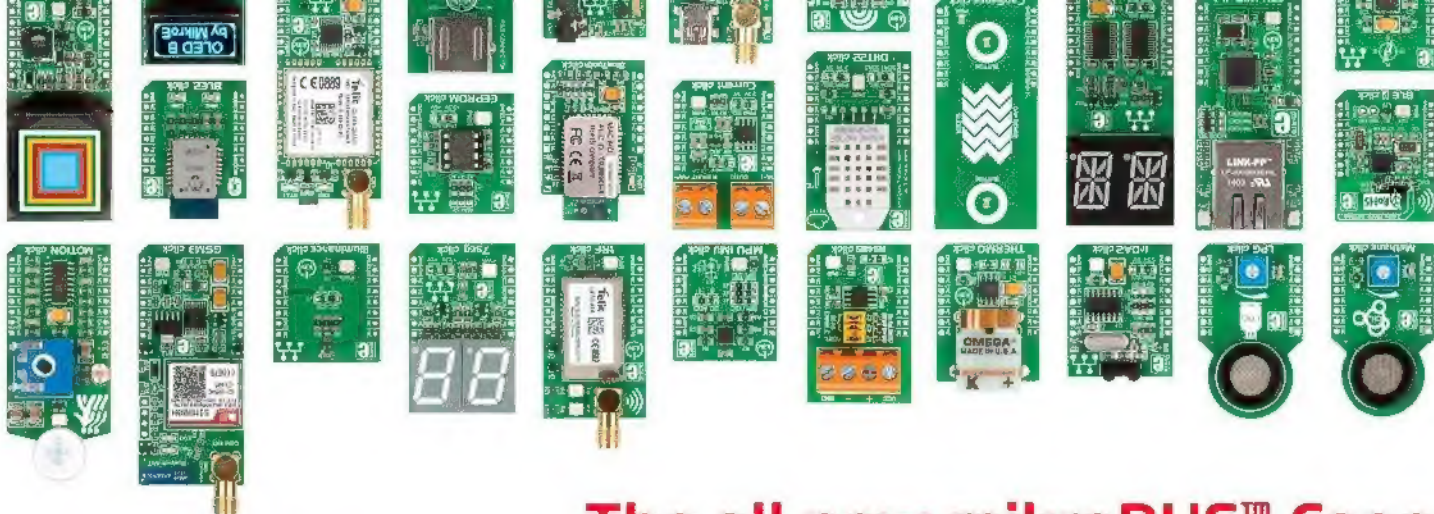
- Dual-channel, 120MHz maximum bandwidth, 20Vpp maximum output amplitude, high fidelity output with 80dB dynamic range.
- High-performance sampling system with 1.2GSa/s sampling rate and 16-bit, not 14-bit vertical resolution. No detail in your waveforms will be lost.
- Sweep and Burst function
- High precision Frequency Counter
- High fidelity, low jitter, 4.3" touch-screen display

## SDG 2000X Series Function/Arbitrary Waveform Generator



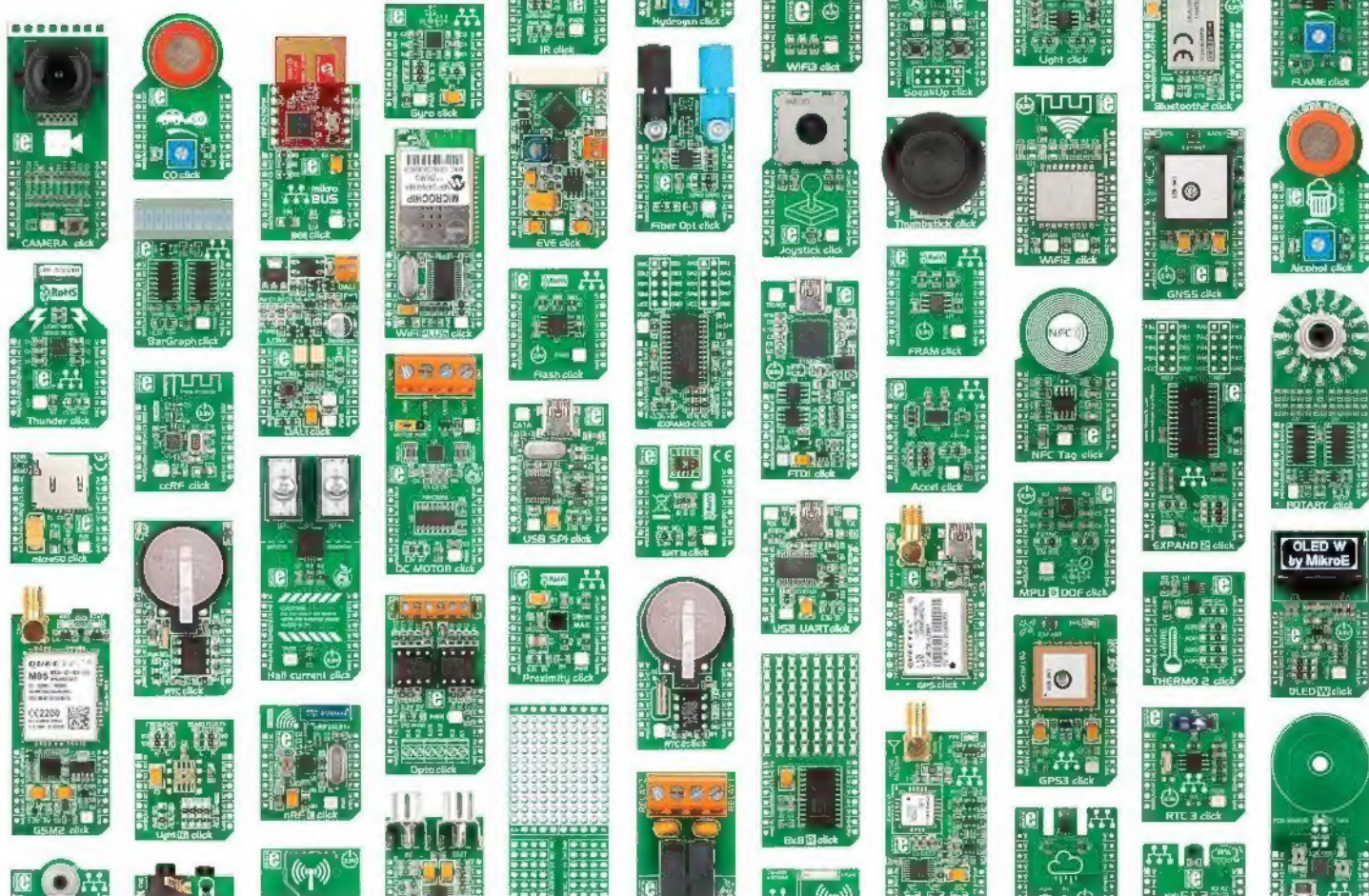
- 3 independent controlled and isolated outputs, 32V/3.2A $\times$ 2, 2.5V/3.3V/5V/3.2A $\times$ 1, total 220W
- 5 digits Voltage, 4 digits Current Display, Minimum Resolution: 1mV/1mA
- Supports panel timing output functions
- 4.3 inch true color TFT- LCD 480 $\times$ 272 pixel display

## SPD 3300X Series Programmable DC Power Supply



## The all new mikroBUS™ Cape

Turn your **BeagleBone Black** into anything you want with the **mikroBUS™ Cape**. With four sockets, you can pick and mix from over **130 click™ boards** to add all sorts of sensors, transceivers, encoders, connectors and many other modules to your projects. **New clicks are coming out weekly, so keep returning for more!**



# October 2015



## 22 Setting Up a Test Bench

What equipment do you *really* need for monitoring and testing all your projects?

■ By Robert Reed

## 31 Understanding Harmonics Using Simulation

Harmonics form a base line for testing, comparing, and explaining various circuits. This short tutorial shows how you can systematically look at the structure of complex circuits using a simulation program and building block approach.

■ By Richard Agard

## 36 Breaking the Arduino Speed Limit — Part 2

This continuation of the March 2014 article on just how fast you can push an Arduino offers a solution to previous clock glitches, adds a serial LCD screen that leaves more pins available, and introduces an eight-bit fast ADC. Then, you put everything together to make an even nicer digital storage oscilloscope.

■ By Bob Davis

## 42 Meet the ESP8266

It's not all that often that a new piece of hardware comes along that immediately captures the attention of the builder community. The ESP8266 is the newest example of this. It's only about the size of a nickel, yet contains a powerful 32-bit microcontroller and a Wi-Fi interface, plus you can buy it for around \$4.

■ By Craig Lindley

## 48 Silent Sensors

Ferroelectric capacitors can help take the mystery out of event detection, and save the results to report on them later.

■ By Joe T. Evans, Jr. and Spencer T. Smith

## 53 Vintage Computing — I Still Adore My 64

Next in our Vintage Computing series is how to emulate a Commodore 64 home computer on your modern PC.

■ By Jim Lawless

## Columns

## 06 TechKnowledgey 2015 Events, Advances, and News

*This time, read about taking another stab at GW detection, a really big tablet, easy cable tracing, an instrument of torture, plus some other cool stuff.*

## 10 The Spin Zone Adventures in Propeller Programming

May the G-Force be with You.

*Badges? We don't need no stinking badges. Unless, they're Parallax's new hackable Conference types complete with a three-axis accelerometer.*

## 16 Q&A Reader Questions Answered Here

*Topics answered include drone mechanics, PID control, and music editing.*

## 60 Practical 3D Printing Real World Uses for the Electronics Experimenter

3D Print Designs for Electronic Hobbyists.

*Discover some handy tools to print that will make a welcome addition to your work bench.*

## 62 Near Space Approaching the Final Frontier

GPSL 2015 and My 150th Near Space Launch.

*Highlights from this year's Great Plains Super Launch conference and commemoration of Paul's 150th adventure into the great beyond.*

## 68 The Design Cycle Advanced Techniques for Design Engineers

From Data Logger to On-Demand Data Storage Device.

*Roland Riegel, Bill Greiman, and the folks at SparkFun laid the ground work for the OpenLog. OpenLog was originally designed as an "out of the box" data logger. We're going to add some PIC32MX electron spice to the OpenLog design and turn it into an "out of the box" general-purpose microSD-based storage device.*

## Departments

05	DEVELOPING PERSPECTIVES	66	ELECTRO-NET
	<i>Home Automation: Are We There Yet?</i>	76	NV WEBSTORE
20	NEW PRODUCTS	79	CLASSIFIEDS
21	SHOWCASE	80	TECH FORUM
		82	AD INDEX

**It's Back! Our Workbench Design Challenge!**  
Details on Page 30!

Published Monthly By  
**T & L Publications, Inc.**  
430 Princeland Ct.  
Corona, CA 92879-1300  
(951) 371-8497

FAX (951) 371-3052

Webstore orders only 1-800-783-4624

[www.nutsvolts.com](http://www.nutsvolts.com)

## Subscription Orders

Toll Free 1-877-525-2539

Outside US 1-818-487-4545

P.O. Box 15277

North Hollywood, CA 91615

## FOUNDER

Jack Lemieux

## PUBLISHER

Larry Lemieux

[publisher@nutsvolts.com](mailto:publisher@nutsvolts.com)

## ASSOCIATE PUBLISHER/ ADVERTISING SALES

Robin Lemieux

[robin@nutsvolts.com](mailto:robin@nutsvolts.com)

## EDITOR

Bryan Bergeron

[techedit-nutsvolts@yahoo.com](mailto:techedit-nutsvolts@yahoo.com)

## VP OF OPERATIONS

Vern Graner

[vern@nutsvolts.com](mailto:vern@nutsvolts.com)

## CONTRIBUTING EDITORS

Fred Eady

Jon McPhalen

Paul Verhage

Richard Agard

Bob Davis

Joe Evans

Jim Lawless

Tim Brown

Jeff Eckert

Chuck Hellebuyck

Robert Reed

Craig Lindley

Spencer Smith

## CIRCULATION DEPARTMENT

[subscribe@nutsvolts.com](mailto:subscribe@nutsvolts.com)

## SHOW COORDINATOR

Audrey Lemieux

## WEB CONTENT

Michael Kaudze

[website@nutsvolts.com](mailto:website@nutsvolts.com)

## WEBSTORE MARKETING

Brian Kirkpatrick

[sales@nutsvolts.com](mailto:sales@nutsvolts.com)

## WEBSTORE MANAGER

Sean Lemieux

## ADMINISTRATIVE STAFF

Re Gandara

Copyright © 2015 by T & L Publications, Inc.

All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *Nuts & Volts Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *Nuts & Volts*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *Nuts & Volts*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: 430 Princeland Court, Corona, CA 92879.

# DEVELOPING PERSPECTIVES

by  
Bryan  
Bergeron,  
Editor

## Home Automation: Are We There Yet?

**W**ith Amazon's general release of the Echo home automation controller, it may be time to take a second look at the home automation market. I first took the plunge into commercial home automation several years ago with X10-compatible hardware ([www.x10.com](http://www.x10.com)).

For the price of a bare-bones Echo (\$180, [www.Amazon.com](http://www.Amazon.com)), you can get a half dozen wireless timers and remotes for controlling lights, appliances, and your home security system. X10-compatible home automation devices are commodities — inexpensive, ubiquitous, and they work. Unfortunately, they're also a bit boring.

At the other end of the home automation market are the cloud-compatible smart thermostats and cameras, typified by the Nest learning thermostat and Nest cam, respectively ([www.nest.com](http://www.nest.com)). Both can be controlled through your smartphone from anywhere in the world. Plus, the learning thermostat is compatible with a variety of devices — from smart locks and sprinkler systems to ceiling fans.

If you want to make the Nest cam fully functional, you'll have to pay a \$10 monthly fee to Nest Aware — not something I'm prepared to do.

One of the advantages of the Amazon Echo is that it's compatible with Belkin WeMo and Philips Hue devices. WeMo is compatible with standard Wi-Fi routers and iOS devices, such as the iPad. Hue — which is primarily for lighting — also works with a standard Wi-Fi router, and both iOS and Android tablets and smartphones.

I've used the Philips Hue lighting system with my iPhone for about a

year. It's expensive, however, at about \$200 for a Wi-Fi/Hue bridge and three 60W equivalent LED bulbs.

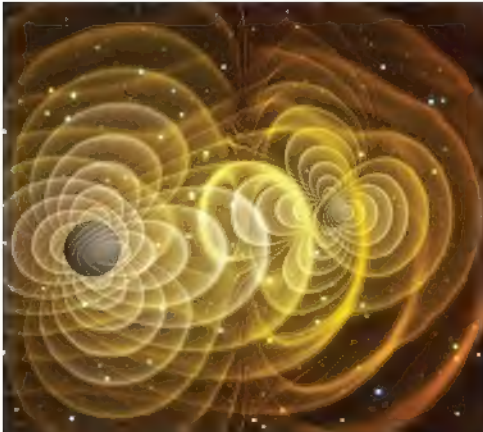
Which brings me to cost. The basic "star trek" package — which allows you to say the equivalent of "Computer, lights on" from anywhere in your living room — is about \$400 — \$180 for the Echo and \$200 for a basic Philips Hue lighting system.

Add a few Belkin WeMo Wi-Fi switches for your existing lights or appliances, and you're easily approaching \$500. Still, this sort of off-the-shelf functionality that actually works was science fiction just a few years ago.

As Google, Apple, and now Amazon compete for the front end of the home automation market, there are likely to be more and more affordable peripherals and tools. More importantly — from an electronics enthusiast's perspective — is the availability of inexpensive peripherals that can be easily torn down and repurposed for other uses. Think of replacing an RGB LED with three opto-isolators to control three servos, for example.

I think we just might have the "star trek" computer system of the 1960s. Now, someone needs to start working on the transporter, so we can say "Beam me up, Echo." **NV**

## ADVANCED TECHNOLOGY



■ 3D visualization of gravitational waves produced by two orbiting black holes. Photo courtesy of Henze, NASA.

### Another Stab at GW Detection

Back in 1916 — as derived from the theory of General Relativity — Einstein predicted the existence of gravitational waves, which are defined as ripples in the curvature of space-time caused by asymmetric (i.e., not spherically symmetric) accelerations of large masses — e.g., supernovae and colliding black holes. Their existence is widely accepted and — as far back as 1993 — a Nobel Prize in Physics was awarded related to observations of binary pulsars that precisely matched predictions of their behavior related to the emission of such waves. The snag is that even though several gravitational-wave detection facilities have been built, none have actually directly detected them so far. The problem is that the amplitude of gravitational waves drops off as the inverse of your distance from the source, so we're talking about extremely tiny perturbations by the time they reach Earth. Your detection device has to be very, very sensitive.

The latest attempt involves two National Science Foundation-funded Advanced Laser Gravitational Wave Observatories (Advanced LIGO, [www.ligo.org](http://www.ligo.org)) in Louisiana and Washington. The facilities are basically laser interferometers in which laser beams are shot down 2.4 mi (4 km) tubes and reflected by mirrors. Any existing gravitational waves should create an interference pattern when the two beams meet, thus proving that they exist. The original LIGO went into operation in 2002 with the capability of detecting changes equivalent to 1/1000th of the size of the diameter of a proton. That didn't do the trick, though, so the Advanced LIGO — with ten times the sensitivity — was dedicated back in May and could be operational by the time you read this.

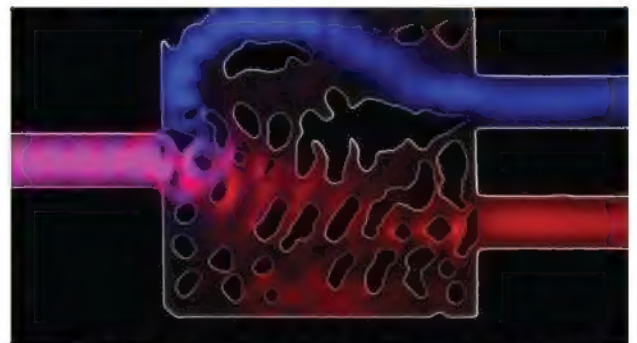
So, what good is all this? Well, if it pans out, the program will allow scientists to look at the last minutes of life of black holes as they spiral toward each other, vibrate, and merge into one larger hole. It will also allow us to test theories of how the universe developed as far back as a nanosecond or so after the Big Bang. If that doesn't float your boat (perfectly understandable), consider that LIGO detects waves in the audio frequency range of 10 to 1,000 Hz, so by feeding the signals into a speaker, you should be able to "hear" them. They are expected to sound something like the audio file located at [www.ligo.org/science/GW-Overview/sounds/chirp40-1300Hz.wav](http://www.ligo.org/science/GW-Overview/sounds/chirp40-1300Hz.wav). ▲

### Algorithm to "Revolutionize" Computing?

Inside your computer, it takes a lot of power to push electrons from one chip to the next. Not only does that waste about 80 percent of your microprocessor's energy consumption (thus generating all that heat), it slows down data transfer through the interconnects. It's not breaking news that photons provide more efficient data transfer, which is why the Internet runs over fiber optic threads. Scaling the concept down to the chip level has been impractical, as optical interconnects are designed one at a time and thousands of them are required for an electronic system. However, a recent *Nature Photonics* article described a new design algorithm developed at the Stanford School of Engineering ([engineering.stanford.edu](http://engineering.stanford.edu)) that could help replace wires in practical systems. The engineers believe that they have broken the design bottleneck with their "inverse design algorithm," which is pretty much descriptive of how it works. The engineers specify what they want the optical circuit to do, and the software provides the details of how to fabricate a silicon structure to perform the task.

"We used the algorithm to design a working optical circuit and made several copies in our lab," noted engineer, Jelena Vuckovic, reporting that the devices functioned flawlessly despite being fabricated in a relatively primitive facility. The Stanford work relies on the fact that silicon is easily penetrated by IR light, and different silicon structures can bend IR in useful ways.

The Stanford structures are so slender that more than 20 of them could fit inside the diameter of a human hair. These silicon interconnects can direct a specific frequency of infrared light to a specific location to replace a wire. The algorithm can create switches or conduits or whatever is required for the task. The Stanford team believes they have set the stage for the next generation of even faster and more energy-efficient computers that use light rather than electricity for internal data transport. ▲



■ IR enters from left and is routed on right at different frequencies. Actual device is the size of a speck of dust. Photo courtesy of A. Piggott.

## COMPUTERS and NETWORKING

### Big Tablet. Really Big.

Tablet displays have been creeping up in size since their introduction with, for example, the current run of iPads offering a 9.7 inch display and a Samsung Galaxy unit offering 10.5 inches. If bigger is better, then check out the nabi Big Tab HD 24" (well, 23.6 actually), offered as the "world's biggest HD tablet made especially for children and their families." The 1920x1080 multi-touch screen lets users play console-quality games (including air hockey, monopoly, and many others) in HD, and room-to-room portability allows you to watch movies, surf the 'net, or play games anywhere in the house. [It weighs 13 lb (5.9 kg), though, so get a good grip.]

The Android-based device can be laid flat or propped up as desired. The unit features nabi's OS 3.0 Blue Morpho which includes more than 400 "kid-focused, parent-approved features." Specs include a 1.6 GHz Tegra 4.0 quad-core processor, 2 GB of DDR3 memory, and 802.11bgn wireless. Don't stray too far from an AC outlet, though, as average battery life is rated at only 30 min.

Accessories include a microphone (with or without



■ The shark game — one of many available on the nabi Big Tab.

karaoke box), headphones, and an HD camera. One quick caveat: Don't let the kids see this unless you're prepared to shell out \$500 (street price) for one. For more info, visit [www.nabitablet.com](http://www.nabitablet.com). ▲

### Time for a New Scanner?

Way back in 1988, Apple introduced the first flatbed scanner, cleverly called the Apple Scanner. While revolutionary for its time, it offered only a four-bit image (16 levels of gray) and 300 dpi maximum resolution. On top of that, a full-page scan took a little over 20 seconds. But, hey, you only had to shell out \$1,799 for one (plus \$795 for Omnipage if you wanted to do OCR work). Today, you can pick up a unit that offers 4,800 dpi resolution, millions of colors, OCR software included, and various image processing features for a lot less. In particular, we're talking about the new Epson ([www.epson.com](http://www.epson.com)) Perfection V39, which allows you to scan, restore, archive, share, and send images directly to the cloud. The price is \$99 MSRP, with street

prices as low as \$79.99. The V39 features convenient front buttons for simple scan jobs, plus a built-in kickstand if you prefer vertical operation. Additionally, the V39 comes with a software package that includes ArcSoft Scan-n-Stitch® Deluxe (allowing you to scan oversized prints and documents), 4 Easy Photo Fix® technology to restore color to faded photos, advanced digital dust correction to remove dust from scanned photos, and OCR capabilities. Okay, I know. You already have a scanner and it still works fine (unless the manufacturer is trying to force you to buy a new one by not upgrading the driver). Just also consider that the V39 takes up only 9.9 x 14.4 x 1.5 in (25 x 36.5 x 3.8 cm) on your desktop, and it is powered entirely through the USB connector — no power adapter needed. It might be worth 80 bucks just for the convenience. ▲



■ The Epson V39: compact, feature-packed, and cheap.

## CIRCUITS and DEVICES

### Cheaper, but Not Cheap

Billionaire Elon Musk — of Tesla Motors, SolarCity, and SpaceX fame — is known for generating intriguing, long-shot business models and finding backers for them, and they just keep on coming. His latest offering — a product spinoff from Tesla Motors — is the Powerwall home battery ([www.teslamotors.com/powerwall](http://www.teslamotors.com/powerwall)) which stores electricity from your home solar panel array in a compact and easily installed package. As noted by the company, "The average home uses more electricity in the morning and evening than during the day when solar energy is plentiful. Without a home battery, excess solar energy is often sold to the power company and purchased back in the evening. This mismatch adds demand on power plants and increases carbon emissions. Powerwall bridges this gap between renewable energy supply and demand."

On the positive side, because the lithium-ion battery is enclosed in an outdoor-rated enclosure, it doesn't take up space in your garage or basement. Plus, it includes a liquid thermal management system, a battery management system, and a smart DC-DC converter for controlling power flow. Perhaps most interesting is that a 10 kWh unit sells for \$3,500, which puts the cost at \$350/kWh of storage. This is significantly better than competing systems which tend to run you closer to \$500/kWh. So, does it make financial sense to install one? Well, consider that the \$3,500 price doesn't include the inverter, installation costs, or possible required upgrades to the home electrical system. Plus, the maximum continuous power draw is only 2,000W, which won't even run two hair dryers at the same time. The bottom line is that a plain old generator is still much cheaper for backup power. Prices are expected to drop dramatically over the next few years, though, so things might become more favorable. If you want a serious analysis now, try out the return-on-investment calculator at [www.catalyticengineering.com/the-berlin-powerwall/#more-605](http://www.catalyticengineering.com/the-berlin-powerwall/#more-605). ▲



■ The Tesla Powerwall on an outdoor mounting.

### Easy Cable Tracing

It can be a real problem when you have a bunch of wires in the house, office, car, or other location, and need to figure out where they go and to what (if anything) they are connected. That's when you need

something like Platinum Tools' new TP150 Tone and Probe set, designed to simplify cable tracing. As explained by product manager George Jang, "With the Platinum Tools Tone and Probe set's steady tone, push-to-scan button, and a clear loudspeaker, you can quickly trace and identify cable locations on jacks or through walls."

The probe features a pushbutton that activates tone scanning, and its tapered tip is designed to allow easy penetration of cable bundles. The tone generator includes a slide switch for selection of tone/off/cont modes, with the continuity mode used to determine if a wire is broken. It includes alligator clips to test wire pairs and an RJ11 connector for data and phone lines. For about \$25, you get both the tone generator and probe, plus a carrying case, 9V batteries, and user instructions. ▲



■ Platinum Tools' TP150 Tone and Probe set.



## CIRCUITS and DEVICES Continued

### Instrument of Torture

This month's milestone in ridiculousness is the Otamatone Deluxe Touch-Sensitive Electronic Musical Instrument ([www.otamatone.com](http://www.otamatone.com)), perpetrated by Japan's Cube Works Company and an art group called Maywa Denki. Shaped like an eighth note and having its sound leaking out of a flexible mouth, you play it by sliding a finger up and down the stem (up for higher notes, down for lower) and squeezing the head to obtain a wah-wah effect. The instrument offers three octave levels which one might describe as high, higher, and insufferable. It does offer a 3.5 mm stereo mini jack, so you can connect headphones, an amplifier, or speakers, but plugging something into the jack doesn't disconnect the internal speaker, so others in the room will still have to hear you play it. The sound is hard to describe, but it is reminiscent of what a balloon produces if you stretch its nose while letting the air out. The MSRP is \$99, but they can be found at the usual Internet stores for about \$25. Probably not something you want for yourself, but it makes an excellent gift for children of people you don't like. ▲



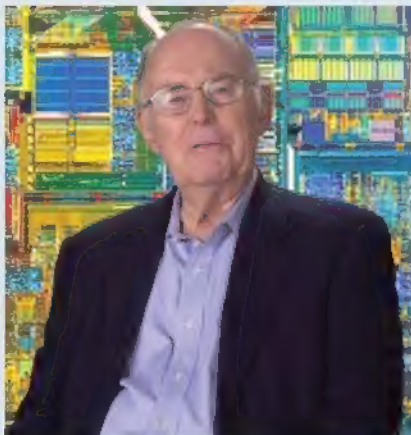
■ The Otamatone is available in several colors.

## INDUSTRY and the PROFESSION

### Happy 50th, Moore's Law

In a magazine article published in 1965, Gordon Moore — co-founder of Intel and Fairchild Semiconductor — observed that, in computing history, the number of transistors in dense ICs had doubled every year (later revised to two years). This observation somehow changed into a prediction for the future and became known as Moore's Law. It has turned out to be correct partly because it was an educated vision, but partly because the semiconductor industry adopted it as a guide to long-term planning and a basis for R&D targets, making it something of a self-fulfilling prophesy.


For years, industry experts have assured us that this can't go on forever, and the doubling will slow down as circuit miniaturization bumps up against the laws of physics. However, Intel says it sees a path to keep the show on the road for another decade or so.



■ Gordon Moore, creator of Moore's Law.

The nice thing is that, unlike many legends mentioned in this space, ol' Gordon is still with us at the ripe age of 86. He's also not among the many scientists and engineers who received little or no financial benefit from their work. Reportedly, his net worth is \$6.7 billion. Way to go, Gordon! **NV**

# PoLabs

For more information or software download please visit 

[www.poscope.com](http://www.poscope.com)

### Connect, Control PoKeys



**USB:** keyboard and joystick emulation

**Ethernet:** web server, cloud data storage, Modbus TCP

**I/O:** digital I/O, 12-bit analog inputs, PWM outputs

**Counters:** digital counters, encoder inputs

**PLC:** graphical programming with free PoBlocks software

**CNC:** 1-8 axis motion controller (Mach3, Mach4 or 3rd party app)

**EASYSENSORS:** I<sup>2</sup>C, 1-wire and analog sensors

**INTERACTION:** matrix keyboard, matrix LED, LCD

**COMMUNICATION:** free libraries for .NET, ActiveX and cross-platform C/C++

### Measure PoScope Mega1+



**ALL IN ONE:** Oscilloscope, Data Recorder, Logic analyzer, Analog and digital signal generator, Protocol Decoder

**PARALLEL** graphics and data processing

**SMALLEST** USB 2.0 portable 2,5MS/s oscilloscope

**DATA ACQUISITION** of analog and digital signals

**DATA RECORDING**

**EXPORT** to CSV, XLS, PCM and HTML

**SIMPLE USAGE** of advanced features

**LOWEST** power consumption- less than 60mA

**FREE SOFTWARE** and updates

# May the G-Force be with You

On a recent trip to Disneyland with my friend, Lynda, she asked me about accelerators, and if I thought investing in a company that will provide them to Apple would be a good idea. Mind you, this is not really a normal conversation for us. I work in tech. She works in financial services. We met in acting class, and most of our conversations center on the arts. Still, it was a fun chat that included my interest in building a portable g-force meter for roller-coaster riders (Lynda and I love the California Screamin' coaster). In the end, I thought her investment idea was sound — especially when one considers the continually-expanding smartphone market, not to mention wearables (e.g., Apple Watch, FitBit, etc.) that require motion and orientation sensing.

Not long after that day, I got a note from my friend, Ken Gracey that Parallax would be creating a Propeller-powered convention badge for anyone who needed such a gadget. The badge we created for

DEFCON 22 was an enormous success, but those are special and were limited (if you want one, sometimes they show up on eBay). For reasons I won't trouble you with, the DC22 badge was a very quick turn-around project, so we kept the circuitry simple. However, with a better schedule, the Parallax Conference Badge is able to include quite a bit more in the way of features:

#### Parallax Conference Badge (Gen 1)

- Propeller Microprocessor
- 64K EEPROM
- 128 x 64 OLED display
- 6 Blue LEDs (Charlieplexed)
- 2 RGB modules (six LEDs, Charlieplexed)
- 7 Touch pads (used as buttons)
- IR output
- IR input
- MMA7660FC three-axis accelerometer
- AV output (composite video, stereo audio)

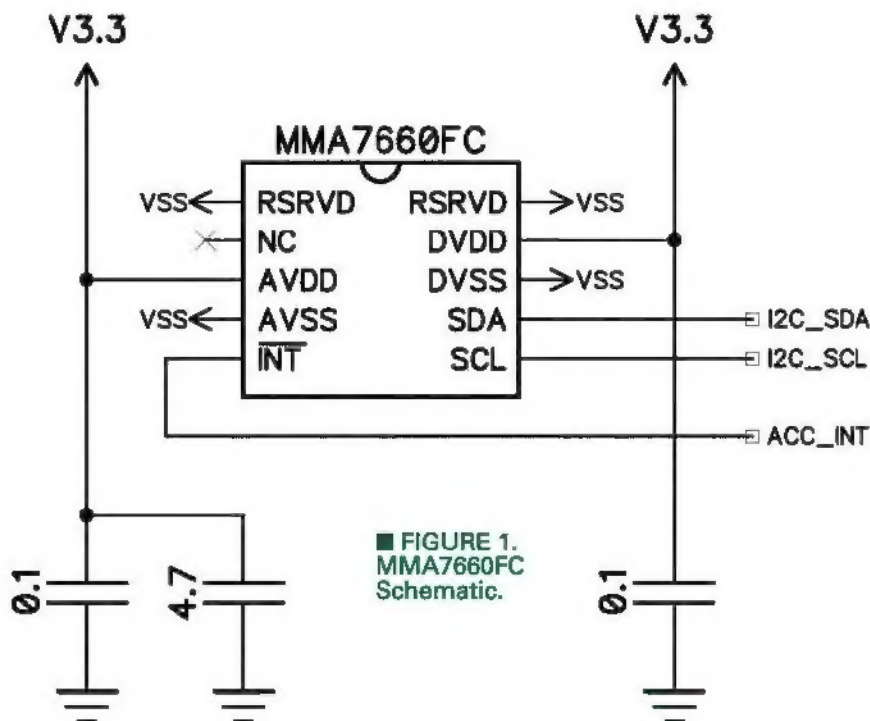
This is a nice list of features for a low-cost (~\$50) convention badge. It's powered by a 3.7V LiPo battery, and even includes a charger circuit. When you plug into a USB port, the battery gets charged.

As you can see, one of the features of the badge is the Freescale three-axis MMA7660FC accelerometer. This is an I<sup>2</sup>C device, so the connections are quite simple. **Figure 1** shows the schematic for the MMA7660FC as used on the badge. No pull-ups are shown for SCL and SDA, but they are on the badge near the EEPROM (the I<sup>2</sup>C buss is shared).

The schematic is simple, but putting one of these dudes together isn't — the chip is tiny: only 3 mm x 3 mm (3 mm = 0.12 inches). Have a look at **Figure 2** — the badge on the left is a pre-production staff badge; the one on the right is a production model guest badge (the production badges have a soft on-off button so you don't have to pull the battery).

The Propeller is easy to find (biggest chip on the board). Above that is the 64K EEPROM. Now, have a look at the tiny chip just above the EEPROM; that is the accelerometer.

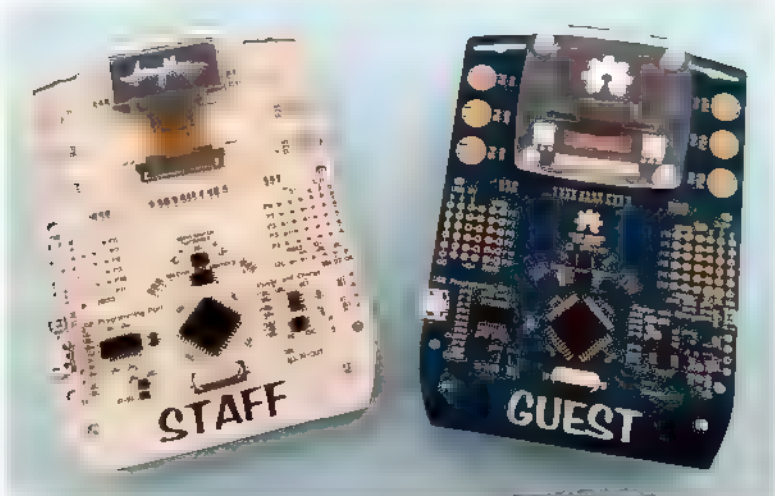
Clearly, this is not something we want to try to solder at home — even if



■ FIGURE 1.  
MMA7660FC  
Schematic.

## ADVENTURES IN PROPELLER PROGRAMMING

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?magazine/article/october2015\\_SpinZone](http://www.nutsvolts.com/index.php?magazine/article/october2015_SpinZone).



■ FIGURE 2. Parallax Conference Badges.

we do toaster-over reflow soldering. No worries! If you don't yet have a Parallax Convention Badge, Seeed Studio makes an experimenter's board for less than ten bucks. There are others, too, and I've listed them in the Resources block. Just remember that the MMA7660FC is a 3.3V device; if you want to use it with a BASIC Stamp or Arduino, you'll need a proper power supply and level-shifters on the I<sup>2</sup>C pins.

The MMA7660FC is designed for devices like cell phones, smart wearables, and gaming controls. At its core, it provides four key pieces of information: X axis g-force; Y axis g-force; Z axis g-force; and tilt status bits derived from the other three registers. It can be configured to control an interrupt pin, but I tend not to use that (an interrupt input is part of the badge design so you can experiment with it).

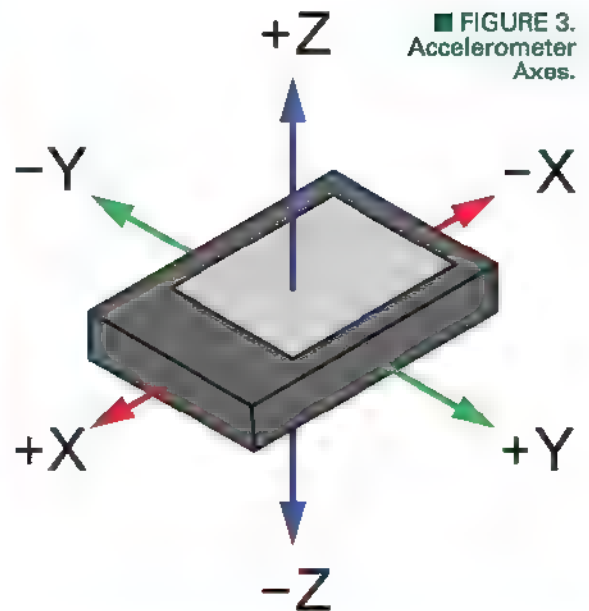
**Figure 3** illustrates the axes behavior of the MMA7660FC. Imagine it installed in a cell phone, and the cell phone laying flat on a table. With the face of the cell phone pointed away from the Earth's core, the Z axis would read +1g (with X and Y at zero). If we stand the phone up so the top is pointed skyward, the X axis would read -1g.

If we now rotate the phone counterclockwise so that the right side (as we're looking at it) is pointed skyward, the Y axis will read -1g. Think of it this way: The illustration shows the behavior of an axis that is pointed skyward.

When not moving, we will tend to see between -1 and +1 on any axis, but the device has a range of -1.5g to +1.5g. If we shake our phone, for example, we can exert more than 1g on it. In fact, a neat feature of the MMA7660FC is a shake alarm; this is set whenever any axis is greater than 1.3g.

The MMA7660FC also does some filtering and can provide "tap" status if we choose. The idea behind a tap is to simulate a button press.

The MMA7660FC is deliberately simple, and working



■ FIGURE 3. Accelerometer Axes.

with it is too. Let's jump in and set it up for typical use:

```
pub start(sclpin, sdapin)
    i2c.setupx(sclpin, sdapin)

    write reg(MODE, $00)
    write reg(INTSU, $00)
    write reg(SR, $00)
    write reg(PDET, $6C)
    write reg(PD, $08)
    write reg(MODE, $C1)
```

We start by connecting to the device via I<sup>2</sup>C. With the Conference Badge, we're using pins 28 (SCL) and 29 (SDA) – the I<sup>2</sup>C pins used by the EEPROM. For stand-alone boards, we can use any set of free pins.

The **write\_reg()** method takes care of the nitty-gritty I<sup>2</sup>C transaction details for writing to a single register. The first thing to do is suspend g-force measurements by clearing the MODE register; this is necessary to set the sample rate. In simple apps, we won't tend to use the interrupt output pin so that register (INTSU) is cleared, too. Clearing the SR register sets the sample rate to 120 readings per second; this rate is required for tap detection.

To enable tap detection, we set up the PDET register; in this case, we're looking for a tap on the Z axis with a threshold of 12. The PD register sets the tap debounce count. You may have to experiment with this value based on your application and device – I'm using a recommendation from a Freescale app note and it seems to work well when I tap the badge with my forefinger. Finally, we activate the MMA7660FC by writing a 1 to bit 0 of the MODE register (the other bits configure the interrupt output – when used – for push-pull/active-high output as required by the badge).



■ FIGURE 4. Tilt Register Bits.

Reading g-force registers is as easy as reading from any other I<sup>2</sup>C device. If desired, we can do a multi-byte read to capture all axes and the tilt register with one call. Here's that code:

```
pub read all raw(p axes)
  repeat
    i2c.start
    i2c.write(MMA7660 WR)
    i2c.write(XOUT)
    i2c.start
    i2c.write(MMA7660 RD)
    byte[p axes][XOUT] : i2c.read(i2c#ACK)
    byte[p axes][YOUT] : i2c.read(i2c#ACK)
    byte[p axes][ZOUT] : i2c.read(i2c#ACK)
    byte[p axes][TILT] : i2c.read(i2c#NAK)
    i2c.stop

    ifnot (long[p axes] & ALERT_XYZT)
      quit
```

If you've worked with I<sup>2</sup>C, you'll recognize this as a very standard multi-byte read transaction. The key to this method is that it requires a pointer to a long, and this long will be used to hold the four byte registers read from the MMA7660FC. The reason for doing this is at the end of the loop. Each of the registers uses bit 6 as an alert flag; when this bit is set, the MMA7660FC may have been updating the register while the I<sup>2</sup>C read was in process, and the register value should be considered corrupt. By storing each of the axis bytes in a long, we can test for any alert flags with one statement using ALERT\_XYZT (\$40404040).

If there are no error bits, we return to the caller. The axes registers hold a signed six-bit value that we need to fix up and convert to g-force before putting it to use. That's easy using a couple Spin operators in this method:

```
pub raw to gforce(raw)
  return (raw << 26 ~> 26) * 469 / 100
```

The first part of the code (in parenthesis) converts a six-bit signed byte to a 32-bit signed long by left-shifting to move the raw sign bit to bit 31, then doing a shift arithmetic right (~>) to realign it. With a standard right shift, we pad the MSB with zero; with the SAR, we pad the MSB with what's in bit 31 (sign bit for longs). This process correctly preserves the sign of the original value.

Finally, we convert to 1/100ths gs by multiplying the raw value by 4.69. Of course, the Propeller uses integer

math, so we do that with a multiply and divide. The tilt register contains orientation information and status bits for tap and shake. **Figure 4** shows the configuration of the tilt register. Using the default initialization, we can call the **side()**, **orientation()**, **tap()**, and **shake()** methods to extract information from the tilt register:

```
pub side(tbits)
  if (tbits < 0)
    tbits : read tilt

  case (tbits & %11)
    %01 : return FRONT
    %10 : return BACK

  return -1

pub orientation(tbits)
  if (tbits < 0)
    tbits : read tilt

  case ((tbits >> 2) & %111)
    %001 : return LEFT
    %010 : return RIGHT
    %101 : return DOWN
    %110 : return UP

  return -1

pub tap(tbits)
  if (tbits < 0)
    tbits : read tilt

  return (tbits & TAP BIT) >> 5

pub shake(tbits)
  if (tbits < 0)
    tbits : read tilt

  return (tbits & SHAKE BIT) >> 7
```

Note that these methods allow you to pass a previous tilt register reading for conversion. We don't have to, though; by passing a negative value to either of these methods, the tilt register will be read and the desired information returned.

The **side()** method tells us the direction that the front of the badge (or our cell phone) is facing. When laying flat on a table with the face pointed skyward, **side()** will return FRONT. If we flip it upside-down so the front is pointed toward the Earth's core, **side()** will return BACK.

The **orientation()** method gives up portrait or landscape mode information when the device is standing up. If the top of the badge is pointed skyward, we will get UP. If we rotate it so the top of the badge (as we're looking at it) is pointed to the left, orientation will return LEFT. Using the orientation bits is how your cell phone knows how to update the screen when you rotate it.

Okay, let's put this together for a little demo as shown

in **Figure 5**. The `main()` method gets things started and waits for us to open a terminal and press a key. After the key press, the terminal is cleared and the report output set up. With the report printed, we drop into a loop that reads and displays data from the accelerometer every 100 ms:

```
pub main : t
    setup
    term.rxflush
    term.rx
    term.tx(term#CLS)
    term.str(@Display)

    t : cnt
    repeat
        update display
        waitcnt(t +
            (MS 001 * 100))
```

The `update_display()` method uses the `acc.read_all()` method to read and convert all of the accelerometer register to g-force values. The rest is simple cursor moving and data display:

```
pub update display
    acc.read all(@xg)
    show gforce(15, 4, xg)
    show gforce(15, 5, yg)
    show gforce(15, 6, zg)

    show side(15, 8, acc.side(tilt))
    show orientation(15, 9, acc.
        orientation(tilt))

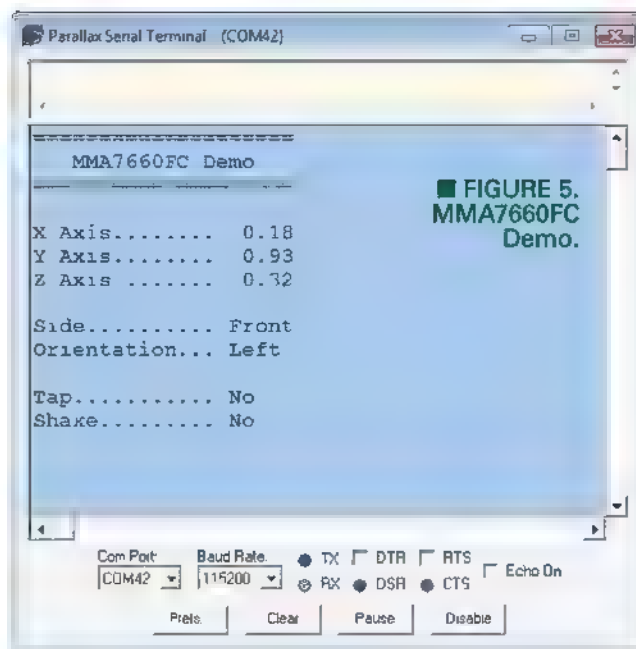
    goto xy(15, 11)
    tf str(acc.tap(tilt), @Yes, @No)

    goto xy(15, 12)
    tf str(acc.shake(tilt), @Yes, @No)
```

Note that `read_all()` expects a pointer to an array of longs. With Spin, we have the option of declaring an array or — as I prefer here — simply declaring four longs (in the correct order) and pointing to the first, like this:

```
var
    long xg
    long yg
    long zg
    long tilt
```

If you look at the g-force readings in the display, you'll see that the Y axis is near +1g. Now, refer back to **Figure 3**. If we took the cell phone in **Figure 3** and oriented it so the +Y axis was pointed skyward, the top of the cell



■ **FIGURE 5.**  
MMA7660FC  
Demo.

phone would be facing left — just as the orientation bits tell us. Neat!

If you tap the badge with your finger, you'll see the tap indication. Don't be alarmed if you see shake at the same time, albeit briefly. Remember that the MMA7660FC sets the shake bit when the g-force on any axis exceeds 1.3g. Actually shaking the device sets the shake indication without setting tap — tap is a quick action and is filtered out when shaking occurs.

## Time to Review Propeller Timing

The last couple episodes have had me talking about a fun little timing object I wrote that just seems to get more useful every day. Part of this has to do with the way the Propeller does timing versus what we may have been used to with a BASIC Stamp or other processor.

For delays, the Propeller uses a command called `waitcnt` (pronounced *wait count*), and I think it's the name that confuses people; a more descriptive name would have been `wait4cnt` (*wait for count*). What confuses the newcomer is that `waitcnt` doesn't wait a specific number of cycles. It waits for a specific value to show up in the `cnt` register. While tricky to understand at first, this behavior is advantageous, allowing us to create loops that run at a fixed time and — if we want — generate a fixed delay like we used to do with `PAUSE` in the Stamp.

If we want to do an old-school fixed delay — which starts now — we code it like this:

```
waitcnt(cnt + delaytix)
```

In this case, the code takes a snapshot of the `cnt` register and adds the delay `tix` to that value (there are minimums for `delaytix` in Spin and PASM). Now, the code will wait until the value in `cnt` reaches what was just calculated. The equivalent code in PASM looks like this:

```
mov    t1, cnt
add    t1, delaytix
waitcnt t1, #0
```

An important point to remember is that reading the `cnt` register takes a bit of time and does pad the timing just a bit. In many cases, this won't cause a problem, but a recent discussion in the Parallax forums points out that accessing the count register in a loop can extend timing in

a sneaky way. To illustrate, a forum member had a bit of code that looked something like this:

```
repeat
  waitcnt(cnt + clkfreq)
  if (++seconds == 60)
    seconds : 0
```

He started his program and synchronized with another timing device. A few days later, he noticed that his Propeller clock was off by more than one would expect for crystal accuracy.

Here's the problem: The way the loop is constructed, the **waitcnt** itself takes a fraction more than a second; the interpreter is reading the **cnt** register and the **clkfreq** register, adding them, and then running **waitcnt**. It's microseconds per iteration, but after enough runs through the loop this extra time will become noticeable.

We're not done. It takes time to read the **seconds** variable, increment it, compare it to 60, and then change it to zero if necessary. Finally, the code has to jump back to the **waitcnt** line. All of these things pad the loop, causing it to run fractionally longer than one second. The fix is very easy, and only requires one variable (a long):

```
t : cnt
repeat
  waitcnt(t + clkfreq)
  if (++seconds == 60)
    seconds : 0
```

This is a very small change with a very big impact. This version of the code will update the **seconds** variable dead-on every second (plus or minus the crystal accuracy). Here's why: Before we enter the loop, we establish an initial sync point by reading the **cnt** register there (note that this is the only place where the **cnt** register is read). The **waitcnt** instruction updates the sync point variable **t** (note the += operator) with the number of ticks in one second (**clkfreq**). When that value is reached, the code falls through to dealing with the **seconds** variable. After that, we go back to the top of the loop and update the sync variable again while doing the next **waitcnt**.

Let's look at some numbers so this is absolutely clear. We'll imagine that we got very lucky when reading the **cnt** register into **t**; it was 0. Now, we add **clkfreq** and the sync point becomes 80,000,000 (for a typical system). The next time through, the sync point will be set to 160,000,000, and so on. The **waitcnt** delays are all fixed from the line before entering the loop; hence, the loop timing stays solid.

We call this a *synchronized loop*, and it will run at the desired speed as long as the code inside the loop consumes less time than the **waitcnt** is set for. If we violate the **waitcnt** timing, our loop will appear to hang because **waitcnt** has to wait for the 32-bit **cnt** register to go all the way around to the original target — this can take about 54 seconds in an 80 MHz system.

I tend to use loops like this when I absolutely need

the loop to run at a fixed frequency — things like motor PWM or servo drivers are good examples. When things are a little more forgiving, I tend to use my timer object and fix any variations using the **adjust()** methods

I have a friend named Brian who is a prop builder in the Halloween industry, and he recently asked for help with a count-down timer for a zombie escape room. This was the perfect application for my little timer object. Brian is not the only person requesting count-down; I helped another guy with a paratrooper training device, and my friend, Matt with a timer for a Propeller-powered arcade game he created.

My timer object calculates the difference between successive reads of the **cnt** register and accumulates milliseconds. So, how do we count down? Simple: We preset the milliseconds to a negative value. Internally, the timer is counting up toward zero; by removing the sign, we have a value that appears to be counting down.

In my timing-oriented projects, I tend to have a method like this:

```
pub reset time
  if (state == S_HOLD)
    time.hold
    time.set secs( TIME_SET)
    last : negx
```

In the escape room and paratrooper training programs, there are hold and run modes; I only allow the time to be reset when the clock is not actively running. When that's the case, I put the timer on hold, preset it to the negative value of the starting time (**TIME\_SET**), and then set a flag (**last**) to a value that differs from the initial time setting.

When starting the timer, I ran into a problem: Displaying the time in whole seconds caused the display to immediately drop one second. The reason is that the timer object does everything in milliseconds. Let's say we preset the timer to -30 seconds and release it to run. One millisecond later, the time will be -29.999. When we divide by 1,000 to get seconds, we get -29 — this makes our display jump time right at the beginning.

Here's how I solved that problem:

```
pub check time | nowms
  nowms : time.millis
  now : -nowms / 1000
  if (nowms // 1000)
    ++now

  if (now <> last)
    show time
    last : now

  if (now == 0)
    state : S_HOLD
    time.hold
```

We start by capturing the current value of the timer

(milliseconds). The next step removes the sign and extracts the seconds from it. Now, the key: We take the modulus of 1,000 from the milliseconds register and if not 0, we bump the value in *now*. If *now* differs from *last* (both are global variables), we update the time display and then copy *now* into *last*. Finally, we check the value of *now* to see if the timer has expired; if it has, the program state is changed to *S\_HOLD* and the timer is put on hold.

Before closing, let me share another trick I do with the timer — again, I used this with Brian's zombie escape room and the paratrooper training device. Each of them has buttons that need to be read frequently, and after a button is pressed, we need a hold-off period to prevent a quick back-and-forth state or value changes. For those programs, I did something like this:

```
pub check buttons | btms
  if (btimer.millis < 0)
    return
  else
    btimer.set(-BTN_SCAN)

  btms : read buttons
  if (btms <> %000000)
    process buttons(btms)
    btimer.set(-BTN_HOLDOFF)
```

For the buttons, I create a separate timer object called *btimer*. The *check\_buttons* routine gets called without delays from the main loop. When the milliseconds value of *btimer* is zero or positive, it is okay to scan the buttons. When this is the case, we set the next available scan time using *BTN\_SCAN* (a constant I tend to set around 25). Now, we scan the buttons.

If a button was pressed (I favor active-high inputs), then we send that value to *process\_buttons* and change the *btimer* milliseconds to the negative value of *BTN\_HOLDOFF*. This will cause the

**Item**  
Electronic Convention Badge  
(guest)

MMA7660FC PCB

**Source**  
Parallax #20100 (staff), #20000

[www.seedstudio.com](http://www.seedstudio.com)  
[www.madscientisthut.com](http://www.madscientisthut.com)  
[emartee.com/product/42141](http://emartee.com/product/42141)

**BOM**

buttons to be ignored for the hold-off period.

In one project, I moved the hold-off setting to the buttons processor method so that I could handle buttons differently. For example, one of my projects lets me modify the time when the timer is in hold mode. When I'm modifying the time, I want a hold-off time of about 250 ms. When I change modes, however, I want a longer hold-off; typically 1,000 ms.

Some will wonder why I'm going through all this trouble. It boils down to this: When our code is sitting on a *waitcnt* statement, we're burning up processor cycles that could be used to do something else. Using the timer object — where appropriate — lets me take advantage of those precious processor cycles.

Until next time, keep spinning and winning with the Propeller! **NV**

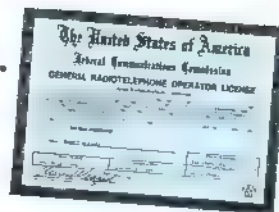
**Make up to \$100 an Hour or More!**



**Be an FCC  
LICENSED  
ELECTRONIC TECHNICIAN!**

**Get your "FCC Commercial License" with our proven Home-Study Course!**

- No costly school. No classes to attend.
- Learn at home. Low cost!
- No previous experience needed!
- **GUARANTEED PASS!** You get your FCC License or money refunded!



Your "ticket" to thousands of high paying jobs in Radio-TV, Communications, Avionics, Radar, Maritime and more.

Call for **FREE** info kit: **800-877-8433** ext. 109

or, visit: **www.LicenseTraining.com**

**COMMAND PRODUCTIONS • FCC License Training**  
Industrial Center, 480 Gate Five Rd., PO Box 3000, Sausalito, CA 94966-3000



RESOURCES

Jon "JonnyMac" McPhalen  
[jon@jonmcp halen.com](mailto:jon@jonmcp halen.com)

Parallax, Inc.  
Propeller boards, chips,  
and programming tools  
[www.parallax.com](http://www.parallax.com)

In this column, Tim answers questions about all aspects of electronics, including computer hardware, software, circuits, electronic theory, troubleshooting, and anything else of interest to the hobbyist. Feel free to participate with your questions, comments, or suggestions. Send all questions and comments to: [Q&A@nutsvolts.com](mailto:Q&A@nutsvolts.com).

- Drone Mechanics
- PID Control
- Music Editing

## Drone Mechanics

**Q** I prefer using the Raspberry Pi. I considered building a vertical lift drone, but they don't have any moving parts. So, how do you maneuver it? Can you show me a circuit for controlling the motors on a drone with a Raspberry Pi?

**Timothy Harner  
Beaver, UT**

**A** The Raspberry Pi Forum is a good place to start looking for quadcopter projects that have been created by others, to get ideas for your own project. Programming for the Raspberry Pi is beyond the scope of Q&A, but the programming for a quadcopter project may already be available at the forum. I have listed the aforementioned website in Q&A SIDELINES.

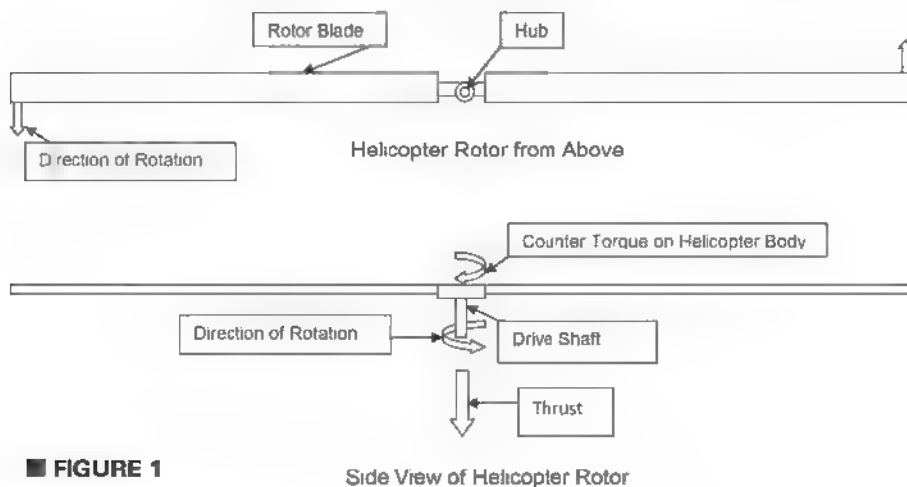
For this answer, I will define a drone as a remotely controlled (i.e., pilotless) aerial vehicle. Drones can be anything from the General Atomics MQ-1 Predator used by the military to a simple quadcopter. First, let's look at how a single rotor (propeller in the quadcopter) operates as shown in **Figure 1**. As the rotors turn through the air, the air drag on the rotor blades produces a counter torque which acts in a direction opposite to the direction the rotor is turning. The greater the speed of the rotor, the greater the air drag on the rotor, and the greater the counter torque. This counter torque is compensated for

by a tail rotor or by using counter rotating rotors (the opposite torques counteract each other). The single rotor helicopter changes the angle of the rotor blade via a very complicated mechanical system to allow the helicopter to hover, ascend, and descend. The four propeller quadcopter eliminates this complex mechanical control system by using electronic controls to vary the speeds of the four propeller drive motors to maneuver it.

As to how to control the quadcopter using only the speeds of the four motors which turn propellers, look at **Figure 2**. It shows the direction of rotation for the four propellers in which diagonally opposite propellers turn in the same direction, while adjacent propellers turn in opposite directions. [NOTE: I am using the term "propeller" for the quadcopter because only its speed varies as compared to the helicopter's rotor.] To hover, all four quadcopter motors turn at the same speed (that is IF all of the motors and propellers have the exact same characteristics, usually there will be some trial and error adjustment of the speeds to achieve hovering without turning). To move the quadcopter in a straight line, you turn one of the diagonal motors faster and the opposite one slower to move in the direction of the slower motor (the faster motor lifts the end it is on and the slower motor with less thrust drops its side, which directs some of the copter's thrust toward the center from the faster motor; the torques should remain balanced so the copter does not

rotate). To rotate the quadcopter, turn two diagonal motors faster so that the counter torques rotate the copter in a direction opposite to the direction the faster propellers are turning.

**Figure 3** shows the pinout and a photo of the Raspberry Pi Model B+. The Model B+ has 40 General Input/Output (GPIO) ports that can be used to control your quadcopter, and it has USB capability so that you can download programs easily. I envision using a joystick which has the forward/back/right/left and stick rotation left/right to move the quadcopter in a horizontal plane,



■ FIGURE 1

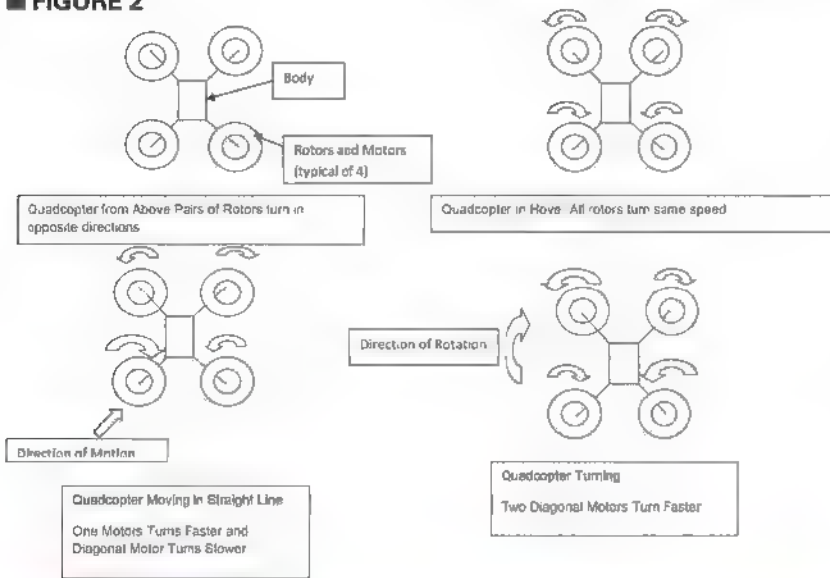
Side View of Helicopter Rotor



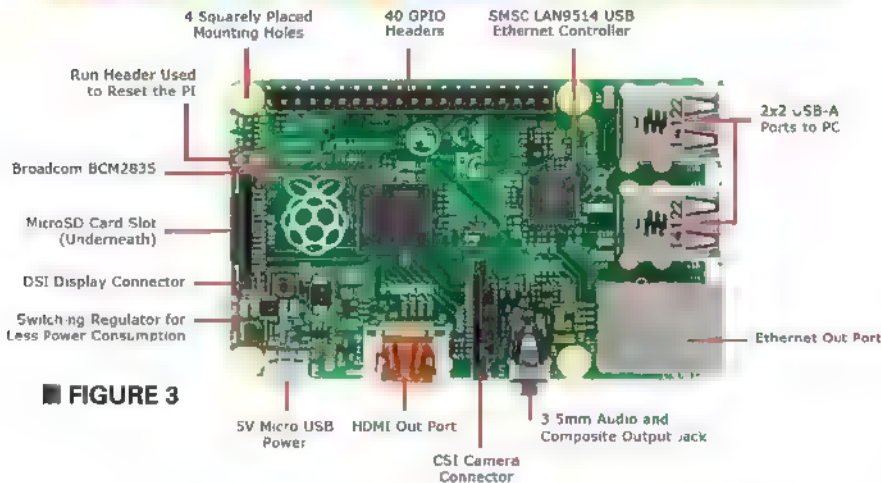
# QUESTIONS and ANSWERS

Post comments on this article at [www.nutsvolts.com/index.php?magazine/article/october2015\\_QA](http://www.nutsvolts.com/index.php?magazine/article/october2015_QA).

■ FIGURE 2



GPIO Pinout Diagram



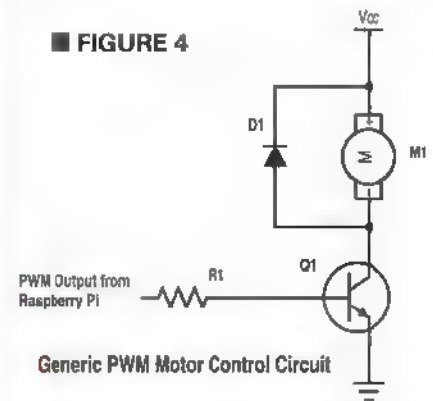
■ FIGURE 3

■ TABLE 1

Operation	Quadcopter Function	Port Configuration
Throttle Forward	Mover Up	Input
Throttle Back	Move Down	Input
Joystick Forward	Move Forward	Input
Joystick Back	Move Back	Input
Joystick Right	Move Right	Input
Joystick Left	Move Left	Input
Joystick Rotate Right	Yaw Right	Input
Joystick Rotate Left	Yaw Left	Input
Servo Motor 1	Motor 1 CW/CCW	Output to Servo Driver
Servo Motor 2	Motor 2 CW/CCW	Output to Servo Driver
Servo Motor 3	Motor 3 CW/CCW	Output to Servo Driver
Servo Motor 4	Motor 4 CW/CCW	Output to Servo Driver

while using a “throttle-type” device to make the quadcopter climb and descend (which is much like the standard helicopter control devices). **Table 1** shows a very rough GPIO map of this control method using eight inputs and four outputs. The Raspberry Pi can send a Pulse Width Modulated (PWM) signal from the ts output port to operate the brushless DC motors which turn the quadcopter propellers either clockwise (CW) or counterclockwise (CCW).

■ FIGURE 4



Generic PWM Motor Control Circuit

The circuit shown in **Figure 4** should work to control the copter’s motors, with the component values chosen based on the motor’s characteristics (voltage, amp draw, etc.). The Raspberry Pi offers the capabilities of “tuning” the Proportional, Integral, Derivative (PID) controls which will make operation of the copter (or any other device being controlled) smoother. See the Q&A answer on “Proportional Integral Derivative (PID) Control” next for PID basics.

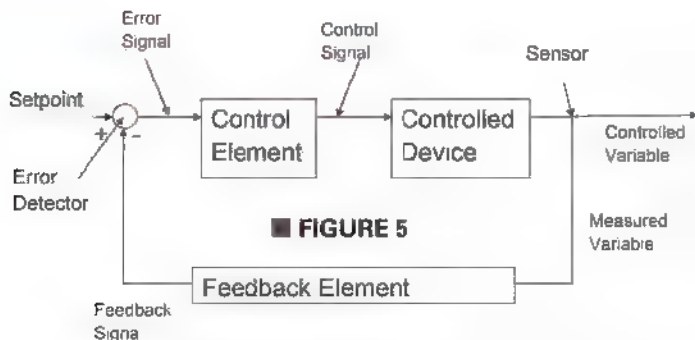
## (PID) Control



Could you explain PID control?



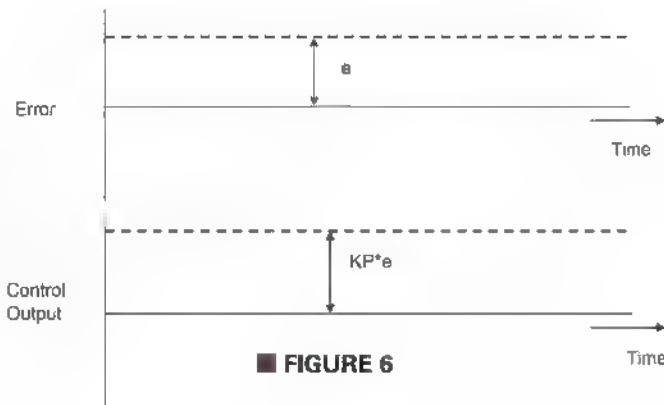
PID control involves a lot of mathematics which take a setpoint (desired operating point) and a measurement of the variable to be



Block Diagram of a Typical Control System

controlled to develop a control output which is supposed to drive the system being controlled to the setpoint of the variable being controlled. **Figure 5** is a block diagram of a typical control system, in which the blocks represent functional units and the arrows indicate signal flow between these units. The setpoint or desired operating value of the variable being controlled (controlled variable) is an input from a human. The measured value of the controlled variable is a representation of the measure of the controlled variable, modified to be compatible with the setpoint value (e.g., if the setpoint is 0 to 5 VDC, representing temperatures from 50 to 250 degrees Fahrenheit, and the sensor is a Type E thermocouple that gives a 3 to 17 millivolt signal for this range; the feedback element must convert between these levels). The error detector subtracts the measured variable from the setpoint and outputs the error signal ( $ES = SP - MV$ ). The error signal is fed to the control element which calculates the control signal using the PID algorithm. The control signal is sent to the controlled device where it drives the controlled variable to the setpoint (if all works as designed).

Now, let's tackle the PID algorithm (mathematical procedure for calculating the control signal from the error signal) by splitting the algorithm into its three components: proportional, integral, and derivative. The proportional control algorithm simply multiplies the error



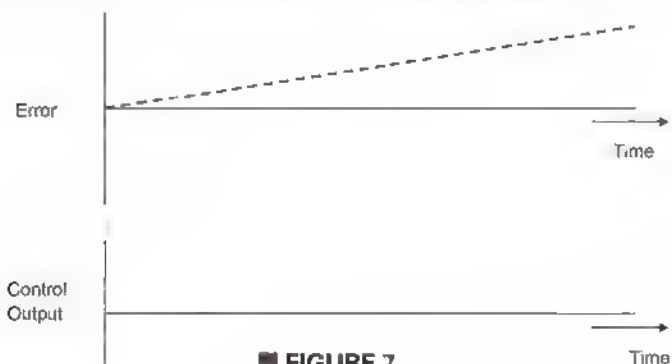
Proportional Control

signal by a constant (proportional gain or  $KP$ ) to determine its contribution to the control signal ( $CS = KP \cdot ES$ ). Proportional control is illustrated in **Figure 6**.

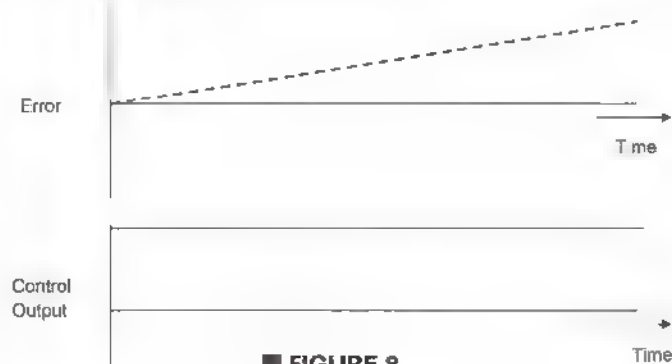
With integral and derivative control, the math gets a little tough, but I will greatly simplify these for our discussion because I know that not all of our readers are that deep into math (see the Internet for a deeper mathematical discussion). In integral control, the error signal is added over a period of time and multiplied by the proportional gain ( $KP$ ) and integral gain ( $KI$ ) to get a control signal that is a function of the magnitude and DURATION of the error signal (as illustrated in **Figure 7**) to prevent the final value of the controlled variable from stopping short of the setpoint due to friction or other resistances in the controlled device.

In derivative control, the control device multiplies the proportional gain ( $KP$ ) times the derivative gain times the RATE OF CHANGE of the error signal to generate the control output which can react to rapid changes in the controlled variable (as illustrated in **Figure 8**).

I will leave combining all of the PID output components for a PID controller to the reader to simplify the current discussion. By tweaking the values of  $KP$ ,  $KI$ , and  $KD$ , we can make the control response perform closer to what we want. Look up these methods of PID tuning on the website listed in Q&A SIDELINES.



Integral Control



Derivative Control

## Music Editing

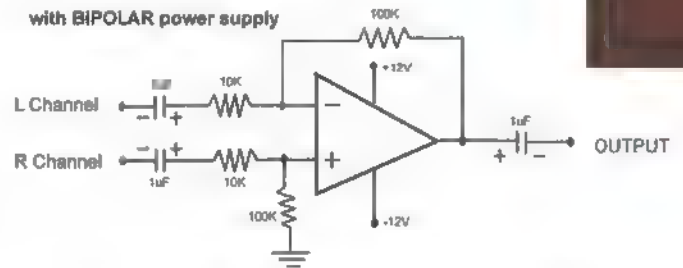
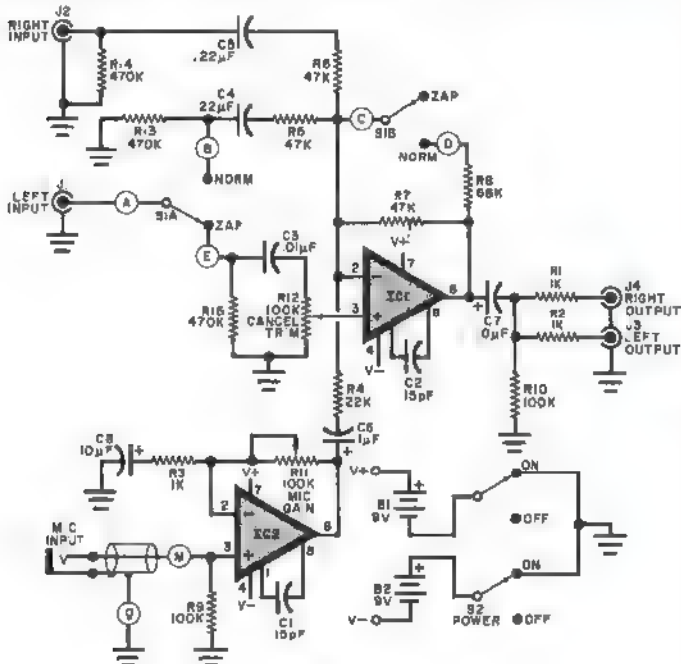
Could you tell me how I can take a song (MP3) and delete the lead vocals to play only the background music? Are there any open source/freeware programs that can do this?

Timothy Harner  
Beaver, UT

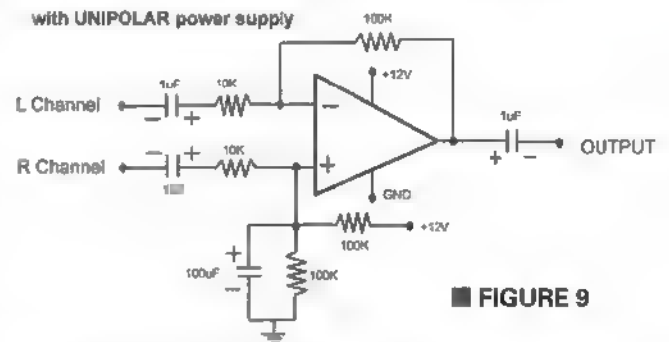
**Figure 9** shows a simple circuit which could be built to remove the vocal signal from an audio recording since the vocals on a stereo recording are recorded equally on the left (L) and right (R) channels. This circuit (using a 741 or TL082 op-amp) merely inverts the L channel so that when it mixes with the right channel, the equal but opposite voice signals are mostly cancelled. The disadvantage of this circuit is its output is monophonic instead of stereo, and it tends to cancel out the bass guitar and drums in the process.

For a better and much more complex circuit, look at **Figure 10**. In this circuit, the time constant of R12 and C3 can be adjusted to eliminate the bass guitar and bass drum signal from the non-inverting input, so these signals are not removed from the final signal. R12 and C3 comprise a high pass filter in which the cutoff frequency is set by the values of the resistor and capacitor [ $f_c = 1/(2\pi RC)$ ]. This circuit has the advantage that it produces a stereo output signal. A disadvantage is heavy reverb which tends to produce weird audio effects, and if the vocal is not perfectly centered between the L and R channels, it will be attenuated but not totally removed.

Making a sound track from recorded music for personal karaoke entertainment is fine as long as you are



⇒ output SHOULD be connected to audio amplifier



■ FIGURE 9

not going to sell the final product. If you are interested in digitally removing vocal signals, I have included a couple of music editing freeware packages in Q&A SIDELINES.

One thing to keep in mind when editing copyrighted music are the copyright laws (see Q&A SIDELINES for information on this. I have been associated with an organization that paid around \$100 to use and display copyrighted lyrics and a \$300 annual rehearsal license to copy sound tracks for members to practice. **NV**

## Q&A SIDELINES

Drone Mechanics

Adafruit

[www.adafruit.com](http://www.adafruit.com)

Raspberry Pi Forums

[www.raspberrypi.org/forums](http://www.raspberrypi.org/forums)

### Proportional Integral Derivative (PID) Control

Nichols Ziegler Tuning of PID Control Loops

[https://controls.engin.umich.edu/wiki/index.php/PIDTuningClassical#Ziegler-Nichols\\_closed-loop\\_tuning\\_method](https://controls.engin.umich.edu/wiki/index.php/PIDTuningClassical#Ziegler-Nichols_closed-loop_tuning_method)

Music Editing

Softonics

<http://vocal-remover-pro.en.softonic.com>

Wavosaur

[www.wavosaur.com](http://www.wavosaur.com)

Audacity

<http://web.audacityteam.org>

Source Forge

[http://mp3.about.com/od/essentialsoftware/tp/Top\\_Vocal\\_Removers.htm](http://mp3.about.com/od/essentialsoftware/tp/Top_Vocal_Removers.htm)

NEW

# PRODUCTS

- HARDWARE
- SOFTWARE
- GADGETS
- TOOLS



## HS-785HB GEAR RACK KIT

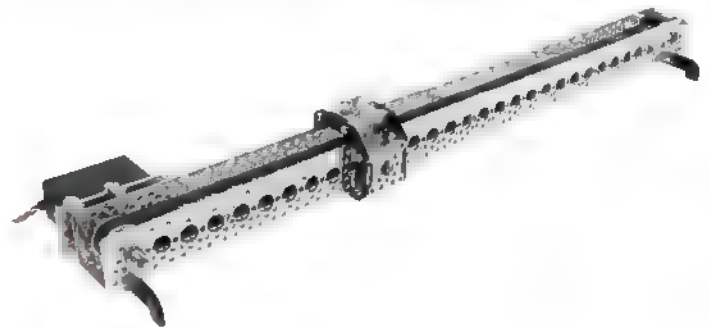
The 785 Gear Rack Kit available from ServoCity is a simple way to create linear motion using a rotational servo. The kit comes with the multi-rotation Hitec HS 785HB servo which allows for up to 9.6" of travel when sending the proper PWM signal from a servo controller. The kit is constructed of 6061 T6 aluminum components and wear-resistant Delrin plastic to create a durable yet lightweight assembly. The framework has several mounting



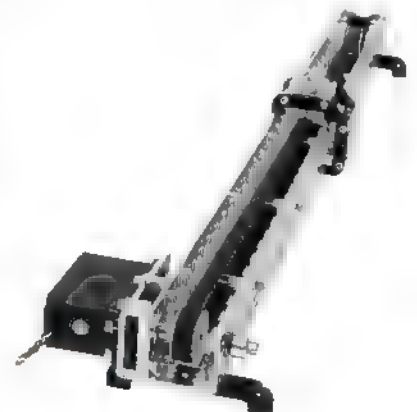
options and integrates seamlessly with the rest of the Actobotics line of products. The 785 Gear Rack Kit is great for steering racks and other applications that require linear motion using a PWM signal. The HS-785HB servo rotates approximately 1.7 degrees/microsecond change in signal. To achieve full travel, a signal range from approximately 850-2150  $\mu\text{sec}$  will be required (each HS-785HB servo responds to a signal slightly differently, so you may need to fine-tune this range in order to achieve full travel without hitting the stops at the ends). On a standard RC transmitter and receiver (assuming a 1050-1950  $\mu\text{sec}$  range) the Gear Rack Kit will travel approximately 6.7". It is priced at \$89.99.

## HS-785HB CHANNEL SLIDER KIT

The HS-785HB Channel Slider Kit from ServoCity also provides an excellent way to create linear motion using a rotational servo. The unique Hitec HS-785HB servo (included in the kit) can rotate multiple turns while



retaining positioning feedback which makes it perfect for this type of application. The XL timing belt and XL pulleys offer a reliable way to transfer the rotation of the servo into a smooth linear motion down the 24" channel backbone. Total usable travel is approximately 19.25" and the rate of travel is .47"/second while running on 6V. Whether you're building a replica, a pick-and-place machine, or a telescoping arm, this kit can help put your



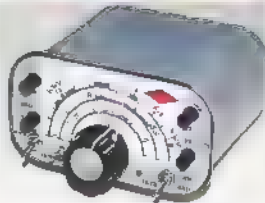
**SKELETONS AND MORE**  
**Chandeliers**  
**Skeletons**  
**Skulls**  
**Sconces**  
[skeletonsandmore.com](http://skeletonsandmore.com)

**NATIONAL RF, INC.**

TYPE  75-NS-3

**MINI HF RADIO**


The 75-NS-3 covers 3.0 to 12 MHz and is offered as a semi-kit.



**CUSTOMER PUTS IT IN A MEAT CAN!!**  
 Visit [www.NationalRF.com](http://www.NationalRF.com) for this and other Radio Products!  
 Office: 858-565-1319

**End of summer clearance sale. Huge savings!**  
  
[www.boxedkitamps.com](http://www.boxedkitamps.com)  
 stereo    headphone    guitar

**PEEK-A-BOO GHOST KIT**  
**PERFECT FOR KIDS, OR ADULTS WHO ACT LIKE KIDS!**  
 COMPLETE KIT INCLUDES:  
 1- PIC-PROGRAMMED PIC16C610 CHIP  
 1- PIC16C610 SKETCH DRIVEN BY 7-SONAR PIR MOTION SENSOR  
 1- BATTERY  
 1- SMALL BROWN MOTOR  
 1- SOUND RECORDING CHIP  
 1- 4OHM SPEAKER  
 1- LED EYE WITH OPTICALLY  
 1- CHINA  
 2- FINGER-TONGUE FEMALE CONNECTOR  
 1- 1/4" JACK  
 1- DONALD ASSEMBLY INSTRUCTIONS  
**Only \$37.95**  
<http://store.nutsvolts.com>

 **QKITS LTD**  
 sales@qkits.com  
**Arduino • Raspberry Pi**  
 Power Supplies  
 MG Chemicals  
 RFID  
**\$8.50**  
 flat rate shipping  
**1 888 GO 4 KITS**  
 Visit us at: [www.qkits.com](http://www.qkits.com)

 **NKE electronics**  
 the MD-8851 training supply, serial cable.  
 Purchase Orders are accepted from all US and Canadian sources.

project in motion. The framework is constructed entirely of Actobotics parts which simplifies connecting additional components to it. Price is \$169.99.

For further information, please contact:

For more information, contact:

**ServoCity**

[www.servocity.com](http://www.servocity.com)

**CAN FD/FLEXRAY OPTIONS ADD TO PROTOCOL ANALYSIS ABILITIES**

**T**eledyne LeCroy has introduced two new trigger and decode options for the WaveSurfer 3000 oscilloscope: CAN FD and FlexRay. The addition of CAN FD and FlexRay trigger and decode capabilities to the WaveSurfer 3000 provide all the tools needed to analyze and debug automotive systems using the CAN FD and/or FlexRay serial data communication standards. New protocol analysis capabilities with CAN FD and FlexRay trigger and decode enable engineers and technicians working with these standards to gain unprecedented insight into their systems, correlating physical layer signals and protocol layer data on a single display. The CAN FD and FlexRay trigger can isolate frame IDs, specific data packets, remote frames,



Continued on page 79

**In the Nuts & Volts Webstore NOW!**  
**NUTS AND VOLTS**  
**NV BOOK SPECIALS**  
 Programming the Raspberry Pi Getting Started with Python  
 Simon Monk  
 Raspberry Pi Three Book Combo  
 Raspberry Pi User Guide  
 Raspberry Pi Projects for the EVIL GENIUS  
**Only \$48.95**  
 Plus FREE Priority Mail Shipping US Only  
 To order call 800 783-4624 or visit: <http://store.nutsvolts.com>  
 Limited time offer.

# Setting Up a Test Bench

Any of you that have followed my *N&V* articles over the past 10 years certainly know by now that I am a test equipment freak. This stemmed from a couple of reasons. Early on in getting my start in electronics and while still in the process of getting my education, I did what many other students were doing at that time – dabbling in radio and TV repair. This not only produced a small income to help with tuition fees, but also gave us invaluable hands-on insight to the actual working conditions of various circuits.

I had also started a small ship-to-shore electronics service business along with the above. With this array of devices waiting for repair, I was literally drooling over high quality test equipment to make my job easier, but school tuition pretty much kept my back to the wall financially. So, I had to do with what I had, or design and make up barebones test gear on my own. Bottom line is that I never lost my love or desire for quality test equipment.

The other reason was that in designing my own equipment or eventually buying commercial equipment, I was blown away by the quality of workmanship and cleverness of design. As far as I was concerned, quality test gear design and construction was the yard stick by which all other circuits were measured.

As time passed, my interest increased further in this area, mainly due to the design challenges it presented and the rewards of completion. A couple of years later and as luck would have it, I was hired by an R&D company that specialized in test equipment design and prototyping. As the old saying goes, “I never had to work another day in my life.”

Now, I am not trying to say that test equipment should be the ultimate goal in your electronics interest. However, once the initial architecture is laid out for any design, test equipment will be required for both the starting point and ending point for those projects. It all comes down to standards and a point of reference (try building a house without the use of a tape measure or level).

Since I have written many articles on test equipment construction, I thought I would back-track this time and look at this from a different perspective: Why do we need TE (Test Equipment)? What variety of TE do we require? What is the best progression of adding TE? And, last but not least, what are the economics of arriving to a fully equipped test bench?

I will address all these issues as sanely as possible based on my 40 years of building, acquiring, and using TE. This article will be mainly pointed towards analog electronics, but much of this spills over to digital use as well. Even with digital design and when all the “number crunching” is completed, it still comes down to analog electronics. This is especially true with ever-increasing speeds now employed – not to mention I/O ports to the real world such as sensors, etc. In reality, there is only one wave shape in the physics of electronics: the pure sine wave – which is analog!

All other wave shapes are an algebraic addition of many sines waves added together versus time. For example, a perfect digital square wave is really derived from a fundamental sine wave and all its odd harmonics (hundreds to thousands) that have decreasing amplitude with each increasing order of harmonic content – again, analog! (Be sure to check out Richard Agard’s article on harmonics in this issue.)

All in all, I can hardly imagine an electronics enthusiast without some sort of test bench. This can be as simple as a folding table and a couple of hand tools, along with a cheap DMM (Digital MultiMeter). Or, it can be as elaborate as a spare room just loaded with TE and a full complement of tools and accessories to go with it. This wide span of facilities depends on which direction(s) your hobby takes you, how deeply you get into it, and the years of involvement.

Interestingly, a lot of top rated engineers in the past all had very elaborate setups right in their own home. Most of them agreed that some of their best ideas originated there due to a quieter and more relaxing atmosphere than the typical workplace. So, location in your own home is something to keep in mind.

So, why do we need test equipment in the first place? Well, for a multitude of reasons, such as initial design and prototyping of the circuits we build; final adjustment and

checkout of the same; troubleshooting various existing devices; characterizing individual components from passives to electromechanical to whatever; even repairing and calibrating the test equipment itself — just to name a few. Without TE, we would be working in the dark most of the time without any standard to refer to.

After acquiring even just basic TE, it will make your hobby enjoyment increase tenfold, and enjoyment is the primary reason we got involved in electronics!

## The Startup Bench

The very first thing to consider is bench location. For many of us, this can be a problem. You will want a quiet location away from heavy foot traffic. This could be a heated garage, basement, or a small spare room, but wherever it's located it should stay fairly dry and without large temperature swings. When space is at a premium, I have seen some pretty clever setups that used a generous sized work bench with tightly fit compartments for parts and equipment storage. When not in use, a wall hinged drop-down table with a cloth skirt attached hid it from view, and then it doubled as a utility table for normal household activities.

Another setup I've seen was a shallow but fairly wide closet in plain view in my friend's den. It was well lit and AC powered. If company arrived, he merely closed its folding doors — out of sight and out of mind. In my first residence, I only had one choice and that was a pass-through foyer between the kitchen and dining room. By using a variation of the above techniques, it kept my wife happy. Then, came our first real home and I divided a fairly large laundry room in two and had a room all to myself, albeit small.

The next two homes I built from scratch. I incorporated that same style for location, with each home having a larger lab area than the previous one to work in. These ended up at the very end of the house, and with easy access to the garage for any accompanying chassis work and such. Except for laundry day, this was the quietest area in the house.

In selecting a test bench, consider a couple options. If you plan on sitting the majority of the time, you will want it at a height of 30". If you'll be standing the majority of the time, you want a height of 36". If you'll be mostly standing but with occasional sitting, you can just use a bar stool.

You can never have enough outlets, so you may want to add a couple of power strips. One word of caution here is do *not* buy the ultra cheap ones as even a moderate load will begin to drop line voltage due to high contact resistance throughout these units.

At least one full length shelf should be constructed over the bench for TE that is common for the majority of projects. The rest can be stored in a cabinet and pulled out when needed. Aside from good room lighting, a

magnifier lamp is almost essential with the increasing flood of smaller and smaller components.

My test bench is a 30" high large metal office desk which I have modified to better suit my needs. I find that I spend most of my time sitting, and this desk does double-duty as a regular office desk for paperwork and such. I then merely set a 20" x 30" piece of laminated hardboard on it to convert it to a work area. The laminate stores very nicely between the bench and wall for quick access.

## The Actual Test Equipment

For your first piece of TE, it goes without saying that it should be a digital multimeter. Your first **DMM** does not have to be top-of-the-line. Plus, you can use some of that savings for future TE. I use a high-end handheld DMM which rarely leaves the bench, but some time ago I purchased some China imports from Harbor Freight for \$3.99 each that included batteries and leads. I use one as an auxiliary meter for the bench and the other stays in the garage for outdoor use. They actually perform quite well. The point here is that entry-level DMMs don't have to cost a fortune.

The next most needed item is one or more **power supplies**. This is one item I would recommend buying the cheapest unit. I will cover this topic and my reasoning on it in more depth later in this article.

I hate to mention **oscilloscopes** this early in the setting up process, but almost every piece of TE from here on in will be greatly enhanced when used in conjunction with a scope. Not that you should run out to purchase one this early, but if the opportunity for a really great buy pops up, jump on it (more on this later).

Probably the next TE after the power supply would be a decent **function generator**. Even a stripped down version with just basic sine, triangle, and square waves and possibly with sweep capabilities, plus any additional features that may be of use to you is fine.

At this point, there is one item that I have found indispensable since I first got into electronics, and that is **resistance and capacitance substitution boxes**. These do not seem to be as popular as they were years ago, and you don't see many around. When you do, the prices are higher, and the quality and ergonomics are lower.

I had bought two of each years ago, and they cover five decades of value. All of them are 1% tolerance and the resistive version uses 1/2W components. Only four or five switches gets you any value in their range in one ohm/pF increments. Many times, it is so much quicker to pop one of these into a circuit and dial to the end result you desire rather than go through a bunch of lengthy formulas to get the same result (timing circuits, etc.). However, you should have a rough ballpark figure of where you want to start.

When I bought these years ago, they were very

popular and their cost was quite reasonable. Today, like I mentioned, the prices have sky-rocketed, probably from lack of sales. As an interesting side note to this, about 10 years ago I had a design project that needed a high powered resistive substitution box. I found one for sale that was somewhat beat up but in working condition. It covered six decades in one ohm steps with a tolerance of 1%, and was rated at 250 watts on any range. I haggled and got it for \$75. A month or so ago, I happened to see a new one advertised for almost \$2,000 (and I thought I overpaid at \$75!).

Next on the list should be a **frequency counter**.

Actually, a step up from this would be a universal counter. This is a traditional frequency counter but also a period counter (reciprocal of frequency). Plus, there are other frequency related functions that extend its usefulness.

I here is a branch-off point, depending on where your primary interests lie. I think an RF generator — even a low grade one — would be an asset no matter which way you go from here. Of course, I have already mentioned the ultimate piece of equipment (the oscilloscope) which I consider to be mandatory. If I was only allowed to own one piece of TE, it would be a good scope. I consider this as necessary as my right hand.

From this point on, the equipment needed will be determined by the direction you desire to go: radio frequency, logic circuitry, digital devices, robotics, and the list goes on and on. Each discipline will have its own specialized requirements for the TE needed.

As we acquire more skills and get deeper into any specific area of electronics, the TE will also follow suit and probably get more expensive. In one's career span, the birth of new technologies will spawn a need for specialized TE that eventually will become useless many years later when *that* technology is superceded by even newer technology (analog TV and CRTs knocked out by digital TV and flat screens, for example). The few essential pieces of TE I have mentioned here will probably be around for a long time, however, as they basically measure the electrical properties of the science of electronics, and that is law.

A though **hand tools** are not TE, they are an integral part of the test bench, so I guess I should touch upon this subject. There are a myriad of tools out there for old and new technologies, with additional ones developed all the time. As to surface-mount, the tweezer style test probes are great, as are the tweezer soldering pencils. Beyond that, everyone has their own favorite gadgets to get the job done easier — even using tools that have no bearing for SMD.

I use a set of old aluminum spring-loaded heatsink clips to hold many of these devices in position for soldering. In more conventional circuitry and through hole, I generally only have a half dozen tools on the bench: a pair of 3" needle nose pliers; a quality flush cutter; a good pair of wire strippers; a few dental picks; and, last but not least, a good surgical hemostat. I use a

fine-tip 35 watt solder iron for just about everything and a bulb type solder sucker from RadioShack, which is the most efficient type of sucker I have ever used (and it was dirt cheap).

In closing out this particular discussion, I have to say that the new "green" solder cannot even come close to the good old Kester 37/63, which just flows and grabs so much better. A good supply of random length clip leads and a few cables terminated in various connectors (BNC, SMA, F, etc.) are also mandatory items. These items will just naturally accumulate as time goes by.

## *Purchasing Commercial Test Equipment*

When it comes to TE, there is just no substitute for quality brand name equipment, and there are dozens of top manufacturers out there. The only piece of TE I would recommend buying as new would be a moderate quality handheld DMM. All other top-of-the-line TE is just way too expensive and out of reach for most of us. So, that leaves us with buying used equipment; the go-to place here is eBay. There are various other websites on the Internet that handle used TE, so don't neglect those. Also, if you are fortunate to live near periodic government auction sales, some incredible bargains can be had — sometimes as low as \$1 per pound if buying by the pallet. eBay does seem to have a corner on the market as to TE sales, so I will dwell mainly on purchases through them. Before I get into specific equipment, I will walk you through a few general guidelines for buying.

First thing I would recommend you do is to *not* buy any equipment that is newer than vintage 1985. This may sound strange to you, but there are good reasons not to. From the late '60s through the mid '80s, there were big design changes in transitioning from vacuum tube to solid-state equipment which rapidly improved over these years. For sure, the '70s and '80s saw some marvelous TE equipment produced by the big name manufacturers, and a lot of it can still compete with newer equipment.

Then, enter the digital age from the late '80s and onward. Older technology was now being replaced with more up-to-date digital designs. Knobs and switches were being replaced by buttons and membrane switch pads. Features and accuracies were scaling upward at an accelerated rate. This was all fine and good — to a point — but had a lot of drawbacks for the home laboratory style of electronics.

Along with all the improvements going sky high, so were the prices. The cost alone for a lot of this TE even on the used market is enough to drive you away. Also, the operation of all this equipment was becoming more complex due to the multitude of features now included. If you do end up purchasing a lot of the newer TE, you will probably end up paying for a lot of features you will never use and/or never understand their use to begin with.



The manuals for the higher end units can be as many as 400 pages, and require more time to read and study. Of course, you will need to refer to the manuals often for the seldom used functions and procedures that slip your memory.

Most will perform just about any operation you can imagine within their intended purpose, but may be slow to set up even for a quick simple test. The worst part of all this is that certain TE is almost impossible for the average (or even way above average) enthusiast to work on.

Even finding part's locations can be very time-consuming. With the ever-increasing board population of SMD parts along with parts getting tinier every succeeding year, your chances increase of causing more problems in the troubleshooting process than the one problem you are originally trying to fix. The traces are so small and close on a lot of these boards that one minor twitch of the test probe can wipe out several circuits. Without "smoke," you will never know it happened, so you are still looking for one problem when, in fact, there are now multiple issues.

Many of these units may have a fairly large footprint and will take up a lot of valuable bench real estate, so all-in-all are better suited to large research centers.

The only occasion I had to repair this type of construction was when I was called to troubleshoot a late mode microwave baseband modem, of which (in older mode s) I was very familiar with. Upon opening the bottom panel, I could not believe my eyes. It looked like the board had been coated with glue, and then a pound of rice pored over it and set to dry. These were SMD parts of the "04" size. I could hardly tell where one part ended and the next one started. I just closed it up and walked away.

My point is that I would never want to own newer TE with this type of construction. If it went sour on me or was that way to start with, I could never make it right. Even in original manufacturing, human hands cannot construct these boards, so that is left up to robotic "pick and place" machines.

Now that I have covered good reasons *not* to buy newer equipment, let me talk about good reasons to buy older TE. For the most part, this equipment will have features and quality specs that will probably be much more than adequate for any project you use it on. They will use through hole or stand-alone parts that are easy to locate, identify, and replace in regards to repair and/or calibration. Although somewhat lacking the accuracy and features of their more modern "brothers," they are simple to use and fast to set up for quick tests. They require no software (or *proprietary* software). In short, they are a much more repairable unit than newer TE.

One caveat here is that even though most of the parts are of the over-the-counter type, some will be difficult or impossible to obtain. Hewlett-Packard may have an occasional "hybrid" part in their units. Hybrids are special circuits that are made up like an IC by using discrete components, and are usually sealed in a metal container, are unrepairable, and unobtainable. Although, the parts do

occasionally appear on eBay for an exaggerated cost.

Some engineering groups will rebuild these for a fee and by possibly trading in the old core. The only other choice is to buy a "junk" unit dirt cheap just to retrieve that part. Certain Tektronix equipment doesn't use hybrids but may have special-run ICs that can sometimes be hard to obtain. These parts rarely go bad, fortunately. Wavetek almost always uses over-the-counter parts. Don't let me scare you off with these remarks; it's just something to keep in mind.

One item I have not mentioned that sooner or later will be essential is the operator/service manual that rarely comes with used equipment. This makes correct usage on the TE so much easier to learn, plus the repair instruction and road maps for locating parts are essential. Also, calibration and proof of performance tests give peace of mind when everything meets factory specs. You can go as far as needed or desired with whatever TE you have on hand to do this. There are some specs that you would need a boxcar of test apparatus for checking, but a lot of them are (for the most part) of minor importance for most of us and not of great concern for everyday use.

These manuals can go for \$25 - \$40 apiece, but there is a website called BAMA (Boat Anchors Manual Association) that has a ton of manuals for free download. You can always research the Internet for any information floating around out there, or check out Internet services that sell manuals on discs for \$5 - \$15. My favorite here is ARTEKMEDIA. They're a little higher in price, but provide excellent reproduction work. I have printed up manuals from their discs that rival the best of the originals.

Now, it's time to start buying. Although I have cautioned you about buying newer TE and sticking with the somewhat older equipment, nothing is engraved in stone here. You may find newer stuff that was made to order for you at a decent price and guaranteed. Also, there is equipment dating back to the early '60s that has excellent specs and a proven track record. The Tektronix "500" series scopes and the Hewlett-Packard "600" series RF signal generators are all solid performers. They are a bit power hungry with their tube design with bulky transformers and, of course, are quite large and heavy by today's standards.

Odds are that any used TE will have some defects. This could be as simple as a blown fuse, burned out pilot light, or an ugly scratch on the front panel, or something more involved on the interior. The TE I have purchased on eBay all had problems, but basically they were minor. A bad power supply, for example, could be lytic, have dirty connections, a bad power switch, or others. All of these are simple fixes to get it working.

Once the unit is powered up and working, I will take a very quick assessment of its operating condition, such as if the mechanical controls are operating okay and if basic functions are at least working. Then, I get busy with a can of contact cleaner and scrub all mechanical parts including accessible connectors.

Barring any needed repairs beyond that, I always go to the limit with a full proof-of-performance check and some calibration adjusting, along with a thorough cleaning and paint touch-up.

These checks will show any other problems that need addressing. For the most part, I find this enjoyable and rewarding work. I mention this so if you do have a minor problem on your purchase, don't be surprised. Many sellers offer a 10 to 30 day return and refund if you're not satisfied with the purchase.

The half dozen TE units I mentioned earlier are the basic items you will want no matter what direction your specific interests take you. When making a decision on any of these units, give some thought to the level of quality you need for current projects and also the level you may want down the road. There is no sense in paying for expensive features you may never use. Of course, accuracy is one important criteria and this subject is worth dwelling on for a bit.

Accuracy without stability is meaningless. I can adjust any given crystal oscillator to a single digit of a PPM (parts per million). Is that accurate? Of course it is! However, 10 minutes later, it may have drifted out of specification; 24 hours later, it may not even be in the same ballpark because of lack of stability. So, when the seller says its accurate, that accuracy has to be defined and may have different definitions for different situations.

For most of our projects, 100 PPM will be more than satisfactory for some parameters and even 10% is good enough for many others. On the other hand, for high level research center projects, any error greater than fractional PPB (parts per billion) might be far too inaccurate. Usually, you can find an accompanying manual to download to check out the factory specs for suitability to your usage.

The general appearance of the TE in available photos can tell a lot about its history. Are there damaged controls on the front panel, large dents on the housing, or a totally grimy and gouged panel? It's probably been abused in its life then.

Beware that it may need more than average repairing. Look at the metal connectors for corrosion and/or deeply embedded dirt and mildew — this most likely means it had long term storage in a damp environment (again, a warning sign of possible circuit board degradation). Some photos show units with so many stickers and labels on them, it gives them a beat-up look. However, these may be calibration, departmental, and company IDs that just accumulated over the years; a careful eye can see through this for assessment.

Read the seller's assessment carefully and take everything with a "grain of salt." Some folks are honest and some are not. One common copout is "we do not know the condition due to lack of apparatus to test it."

At any rate, these are the guidelines I have used in my purchases and I've been quite successful. I have only been stung once due to an unobtainable custom band switch, but I bought the unit for such a cheap price that I kept it

as a part's donor for a similar unit.

Other items may be advertised as "will not power-up." I love these sales as the price is usually vastly reduced. Granted, it could be something as serious as an expensive burned out power transformer, but the odds are it is more likely in the 120V primary section that only needs a quick and simple fix.

After looking over an ad thoroughly and deciding to buy, I determine ahead of time what is the maximum amount I am willing to bid on it. I leave eBay and let it do the automatic bidding for me, so I don't have to play games with other bidders.

Another piece of advice is don't get caught up in a bidding frenzy. It is humorous to see folks go crazy with this. On several occasions, I have seen two identical items that had very comparable images and descriptions. One started at \$100 and the other at \$9.99. The cheaper one attracted an army of bidders, and the bids escalated rapidly. As the auction reached day 3, the frenzy was on and the bidding was up over \$150 while the other unit still sat there at \$100 with no bids. The auction week passed and the initial \$9.99 unit winning bid was \$225. The other unit closed without a single bid — go figure!

Let's touch bases now on the initial items I mentioned that I considered essential, and give you my favorites for each category. Starting with the DMM, although there are many quality ones out there, my favorite has always been Fluke. Aside from being high quality, they just seem to take a beating and bounce right back. Every industrial environment I have been in, Fluke was the meter of choice. When purchasing a DMM — be it bench or handheld — I would not recommend buying one with more than a 4-1/2 digit readout. The reason is that these will suffice for today's lower standard supply voltage (3.3 VDC and 1.8 VDC) powered chips, and still have all the accuracy you would normally need. To go beyond that, they are getting into extreme accuracy and resolution, and require regular calibration checks to maintain any confidence of that high level (which can get expensive).

As to power supplies, I have never bought a commercial supply. Right from the start, I have designed and built my own, beginning with a high voltage vacuum tube model and several lower voltage "stick built" solid-state types. Not long after that, in the early '70s with the advent of the 78xx series of regulator chips and the LM317 ICs, they became one of the simplest and easiest circuits to put together — especially if you had a junk box of salvaged power transformers.

All-in-all, I built 15 regulated supplies ranging from lower voltage/500 watts to 300 volts/30 watts. They have come and gone over the years, and currently I have settled on just four in my lab: 0 to 300 VDC at 100 mA; 12 VDC at 10 amps; dual 0 to 20 VDC at one amp with 5 VDC added; and one small low voltage auxiliary supply.

These have served me well in all my projects in the last 10 years. All but the 12V/10 amp units are shown in **Figure 1**.

They all perform well, look good, and have saved me a lot of money. They're not all that difficult to build, so you can do the same. If you are just jumping into electronics, you may want to buy a simple inexpensive supply just to get you started. Then, you can build your own as needed. One word of caution here. In recent years, linear supplies have gone out of favor due to the more efficient switch mode type of supply. However, when you have AC outlets everywhere, efficiency is not a great concern. The last thing your test bench needs is another source of RFI, so stay with the linear supplies.

When it comes to function generators, Wavetek's from the '70s are my first choice. Entering the '80s, Wavetek merged and changed ownership so many times, it was hard to keep track of just who they were. Unfortunately, the later model plastic case equipment I have used seemed to be of lesser quality. I have had good results with the '70s vintage all metal enclosure units, though.

One of my favorites is the model 111. It is just a basic function generator with sine, square, triangular, and positive or negative pulse outputs — all of which can be externally swept. The high end is only 1 MHz, but it is made in one sweet little package occupying very little bench space. It is accurate and of very high quality construction. I also like the model 114 which has all the above, plus internal sweep capabilities as shown in **Figure 2**. Actually, the model 110 through model 142 are all of equal quality and



**FIGURE 2. Wavetek Model 114.**

construction, with each one having their unique additional features and upper limit frequency generation (10 MHz on the 142). Expect to pay between \$30 - \$180, depending on the model and condition.

Hewlett-Packard radio frequency generators (in my opinion) are the best out there. Their early history is steeped in leading RF products, partly by acquisition of other high-end companies in the field. In this category, my choice is the HP8654B shown in **Figure 3**. It is without a doubt the finest L-C (inductor/capacitor tuned) generator I have ever used or owned.

It covers 10 MHz to 520 MHz in six bands, putting out a beautiful sine wave of up to +13 dBm (about 2.8V P-P). Attenuation is continuously variable all the way down to -130 dBm. It has calibrated AM modulation 0-100% and calibrated FM modulation 0-100 kHz (a rarity on many RF generators). It is the smoothest tuning and most stable of any L-C type generator I have ever used. Also, a proof-of-performance check showed it beat most factory specs by



**FIGURE 1. Lab built supplies.**

a factor of 4.1. There is an adjunct unit (1P8655A) counter/synchronizer that is available for this that nestles nicely right on top of it (the upper unit in **Figure 3**). It phase-locks the RF generator in 500 Hz resolution steps to 1 PPM stability, and also reads out frequency. It's a nice package with only a moderate footprint. However, the down side is that its low frequency cutoff is 10 MHz. Expect to pay \$125 - \$275 for the generator alone, and maybe \$40 - \$100 for the synchronizer unit.

The HP8640 is another good generator similar in specs to the 8654, but it does tune down to 450 kHz. It also has a built-in frequency counter and a larger footprint. Expect to pay \$250 - \$600 for this one. There are add-on units available that tune the 8,654 down to 100 kHz, and then 8,640 up to 1,040 MHz, but are somewhat rare to come by.

While I am on the subject of RF generators, I just have



**FIGURE 3. HP8654B and HP8655A.**



**FIGURE 4. HP8601A.**

to mention the HP8601A shown in **Figure 4**. This is a sweep generator that thinks it is an RF generator. Although most sweep generators don't really have or need the accuracy, stability, or precision attenuators that one would expect in a strictly CW (continuous wave) signal generator, this one does. It has excellent accuracy, stability, and a large range of attenuator settings. There are two band ranges of 100 kHz to 11 MHz, and 1 MHz to 110 MHz. The sweep function is super linear. If this one is acceptable for your upper frequency limit, check it out. Current winning bids on these are \$100 - \$225.

Next up is frequency counters. No particular preference here, as there are scads of them out there. You can't go wrong with HP, General Radio, and other big names. There is probably a counter made to suit any purpose, and if you opt for a universal counter you will



**FIGURE 5. Tektronix 2465.**

gain period counting along with some other neat functions. (Even the worst of counters would be suitable for projects that do not require high accuracy, resolution, or frequency.)

Here, you want to determine your needs as to RF work, audio work, etc., and how far you want to resolve the frequency of interest. I would look for at least a seven-digit display with decent sensitivity and have it extend to at least 100 MHz with no worse than 10 PPM accuracy. I prefer LED readouts, but I use both that and the LCD readout in my lab. Prices are all over the place ranging from \$30 - \$400, depending on the hobbyist's level of work.

Now for the "King of the Hill" – oscilloscopes! My clear cut choice here is the Tektronix 2465 series shown in **Figure 5**, with bandwidths ranging from 300 MHz to 400 MHz. Also, there's the 2445 series with the same features, but a bandwidth of 150 MHz is a nice scope if the lower bandwidth fits your needs. This may be the last of Tektronix' pure analog scopes that have a touch of digital features included, such as cursors to indicate period and amplitude along with the current function settings printed out on the screen. It's very quick to set up and operate, with good sensitivity and a wide bandwidth. It also features two 50 ohm input channels for accurate readings at high frequencies, and is a fairly small footprint for a CRT display scope.

There is one caveat here. There's a history of the X axis (horizontal) sweep driver chip failing. This chip was a special run just for this series, and is now unobtainable. However, this defect may only occur in one out of every thousand units. Expect to pay \$200 - \$500 for the 2465, and \$100 - \$250 for the 2445, again, depending on the condition. I reluctantly paid \$500 for mine but it had been recently calibrated and has performed superbly for me.

Analog scopes have what is known as a Gaussian roll-off in regards to the frequency response curve. This is a gradual and steady drop in display amplitude vs. frequency, occurring a little before the corner frequency (3 dB point). Even though this drop-off occurs, I calibrated my scope on every range and both channels, all the way to somewhat beyond 500 MHz. Now, I just use lookup tables for any frequency displayed, and use a simple correction factor to know exactly what the true amplitude is to the displayed level.

In general, analog scopes will have a perfectly flat amplitude response for about one third of their specced 3 dB bandwidth. This one is no exception, and actually is perfectly flat out to about 120 MHz. I only mention this as something to keep in mind for any scope you decide to purchase.

I have not mentioned the later model flat screen digitized scopes because they can get pretty pricey, although Rigol and Tektronix have recently come out with some low-end models that sell in the \$500 range. These scopes have a pretty flat frequency response within their rating, but follow a different response curve than the

Gaussian noted above. Known as the “cliff effect,” upon reaching the end of their stated bandwidth, the signal rapidly drops off to nothing and does not stretch out the usefulness as the Gaussian would

I have only had occasion to use these newer scopes a couple of times. What I loved about them was the ability to store wave forms in memory for later comparison, and also to display and hold very long period wave forms. However, I felt there was too much clutter and information on the display which — at least for me — bothers my concentration on the one thing I am looking for. Most of the time, I like to see one quick trace and move on.

## *What to Buy and When to Build*

As stated earlier, there is just no substitute for high quality test equipment, but there are many instances where it is very justifiable to build your own. Again, I would definitely recommend you build your own power supplies as needed over time. With today’s chips available for that purpose, superior performing supplies can be built easily and cheaply to exactly suit your own requirements. Other reasons would include the following:

- TE that you cannot afford at the time, but can use a decent “starter” unit.
- TE that you only have need for occasionally, for non-critical tests.
- TE where you only need the very basic functions.
- TE for one specific testing purpose.

The above TE would include, for example, simple function generators of limited high-end frequency; RF generators for limited tests such as a 455 kHz and 10.7 MHz IF generator; basic multi-waveform generator; current sources; and test jigs just to name a few. There seems to be an application specific chip available to get you started for any of the items you would want to build.

Other very doable and useful projects could be timers, period counters, and metered variacs just to name a few. In fact, there are only two items in my basic list that you would never want to attempt: the DMM and the oscilloscope. All the rest can be built to fairly decent standards. There is just a personal gratification in competing a useful and professional looking project that can’t be described.

One point about construction that I consider mandatory is a practical and handsome enclosure and front panel with neat artwork. I see so many DIY projects that never get past the solderless breadboard with leads and controls hanging off of it in a totally ugly fashion. It leaves you guessing what does what. For my money, I would not even start a project like that. A finished and enclosed unit is just so much more enjoyable to use and store that it is definitely worth the extra effort. Since most metal commercial enclosures are expensive, it almost goes

without saying that you should build your own. I always have my eyes open for any kind of enclosure that has been scrapped as it may suit one of my projects.

I once fished a very small toaster oven out of a curbside trash can that I spotted on my way to work. A little metal rework and fresh paint transformed it into a beautiful case for the project I was completing at the time.

More often than not, I end up making one from scratch. I usually use the softer grade of aluminum for its easy bendability. Almost all woodworking tools — especially those with carbide blades — will machine this stuff like a knife going through butter. I use the following thicknesses:

- 0.02” — Shielding and small subassemblies
- 0.03” — Small panels and cover plates
- 0.06” — Front panels, cover plates, and small chassis
- 0.125” — Chassis only

Along with common hand tools and a pop riveter, a small nibbler tool (about \$7 - \$8) comes in handy — especially for square holes. That and a “poor man’s brake” (made from miter-edged 1x lumber clamped into a bench vise as described in some of my previous *N&V* articles) completes my so-called sheet metal shop.

As to TE circuits available, the Internet is full of them, along with numerous magazine articles over the years. You could even peruse past editions of *N&V* under my name for several that I have contributed. I guess the final factor here is not so much what you can afford to buy, but rather what you can justify to buy. This will be the predominant answer in the buy/build decision.

## *Conclusion*

I hope this article has given a better insight on the why, when, and how of test equipment for your bench, and some of the caveats to look out for. I know I may catch a certain amount of flak here for some of my thoughts but it has all stemmed from my real world experience, and that I cannot deny. For some, they may think the older equipment I favor is junk, but this has all proven to be high quality TE, and foremost, it is repairable by most enthusiasts. Granted, the newer TE specifications may far exceed the performance of the prior, but are those higher specs necessary for most of us?

Consider two things here. One is repairs that I am sure even the best of us could not accomplish. This alone could be double the cost of the original purchase price. Two is directed toward highly accurate pieces of TE. Accuracy does not come cheap. Also, the stated high accuracy at the time of manufacture is meaningless down the road without periodic calibration checks — an expensive proposition.

Add in the cost of repair when one of these parameters won’t pull in to spec on its own, and the total cost could be astronomical. **NV**

The Nuts & Volts

# Workbench Design Challenge Redux

## THE CHALLENGE

Design a complete, fully-functioning, *entry-level* electronics workbench for under \$200, keeping in mind that you must do so using regularly stocked items from reliable vendors.\*

Your entry should be submitted in the form of a list with a line for each item, its associated part #, the vendor name, and the item cost. Make sure the total is \$200 or less!

So, do you think it's possible to stock a workbench on such a stingy budget, including tools, test equipment, and the like? If you think you have the magic formula, submit your list of items and prices to [challenge@nutsvolts.com](mailto:challenge@nutsvolts.com). The best entries will be published in *Nuts & Volts* and be featured in our ReallyCoolReads newsletter. And of course, will win a fabulous prize provided by the companies listed below!

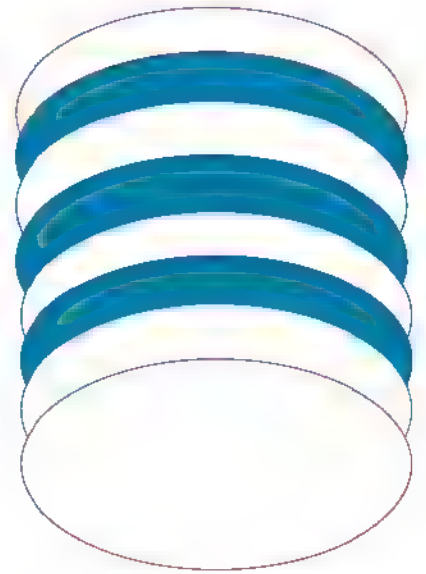
*The prize pool is getting pretty deep  
with great stuff from your favorite companies like  
Parallax, Measurement Computing, Bitscope, Saelig, and Oscium!*

Need more details?

Go to [www.nutsvolts.com](http://www.nutsvolts.com) for the official rules.

**\*NOTE:** To insure a level playing field AND a solid, repeatable design, sources such as eBay, Alibaba, Craigslist, auction sites, or other places that feature very low cost, private party, unknown manufacturers, unknown quality, discontinued, or where stock may become unavailable at any time, should not be considered. When you're done with your entry, it should be usable to create a solid, stable, Bill of Materials with reliable sources. Think in terms of when you're done, it should be possible that your entry could actually be used by a school or university to reliably stock and restock multiple workbenches at a reasonably known cost from reliable and supported sources, for a year or more. Good luck and happy hunting!

# UNDERSTANDING HARMONICS USING SIMULATION



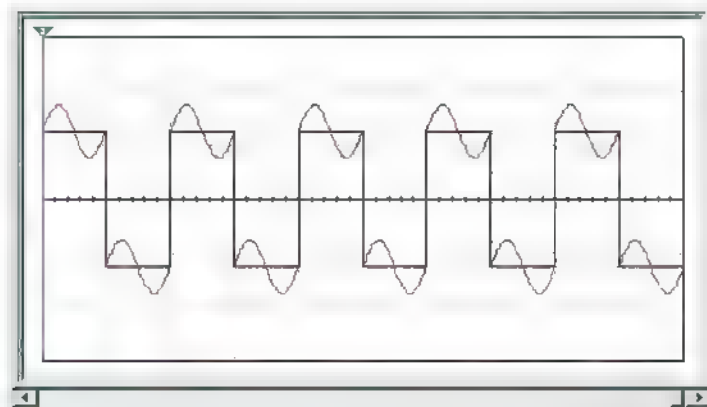
*This article uses simulation to explore harmonics, which are whole number multiples of a base frequency. Harmonics form a base line for testing, comparing, and explaining various circuits. The term can also refer to the ratio of the frequency of such a signal or wave to the frequency of the reference signal or wave.*

By Richard Agard

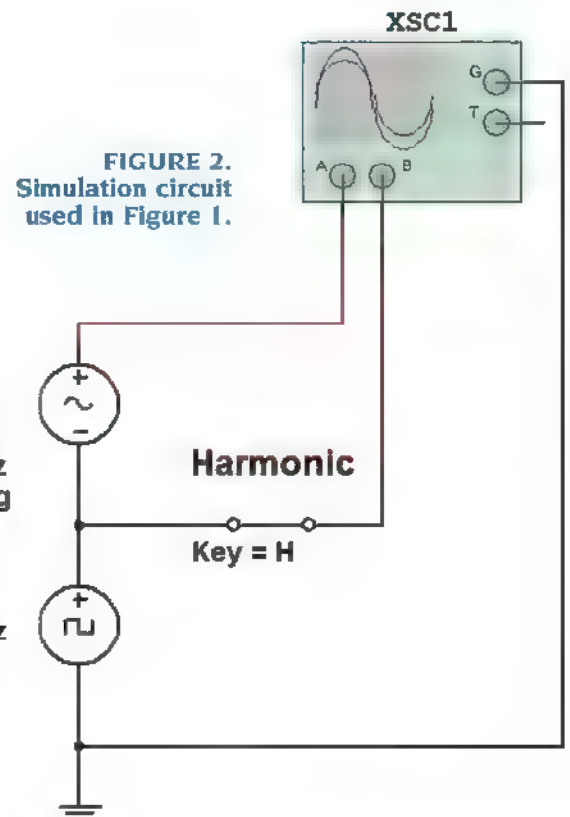
Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?magazine/article/october2015\\_Agard](http://www.nutsvolts.com/index.php?magazine/article/october2015_Agard).

**T**o start off, I simulated a base square wave with a sine wave harmonic at twice the base frequency as in **Figure 1**. I put a square wave source in series with a sine wave source at twice the square wave's frequency and viewed the result with a virtual dual trace oscilloscope. The base frequency is in black and the sine wave is in red.

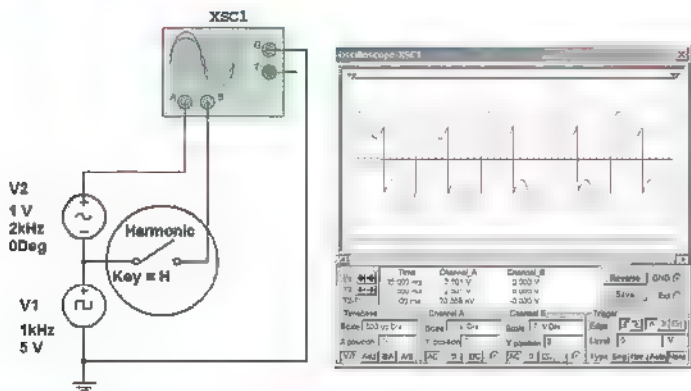
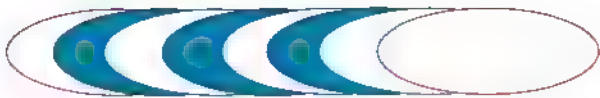
**Figure 2** shows the simulated circuit used to create the wave form in **Figure 1**. A harmonic switch was added so that the enhancement could be switched off and the



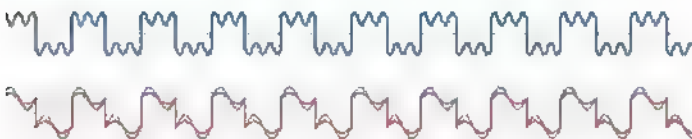
**FIGURE 1.** Enhanced square wave with second harmonic interference.



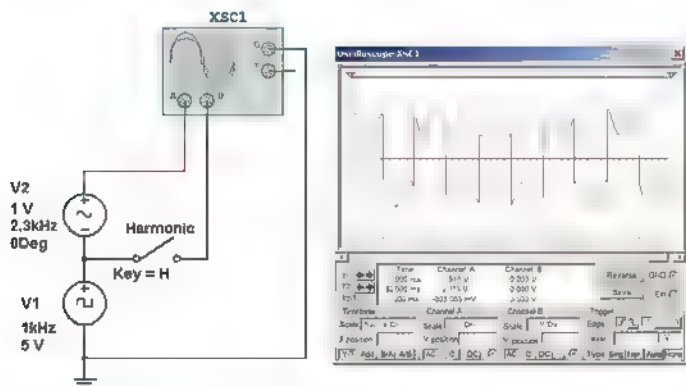
**FIGURE 2.** Simulation circuit used in **Figure 1**.



**FIGURE 3.** Circuit with harmonic enhancement switched off.



**FIGURE 5.** Signal graphic manipulation.



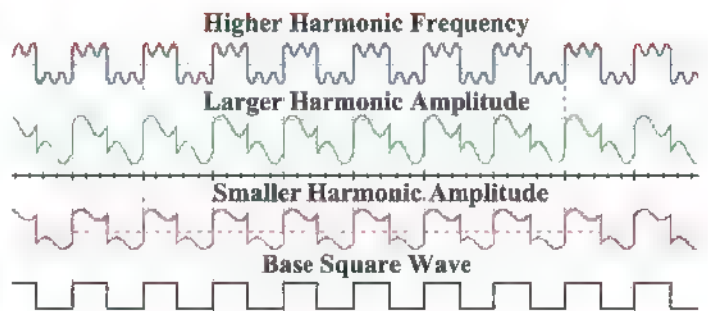
**FIGURE 7.** Square wave with non-harmonic interference.

base frequency waveform viewed. The operation of the harmonic switch is demonstrated in **Figure 3**.

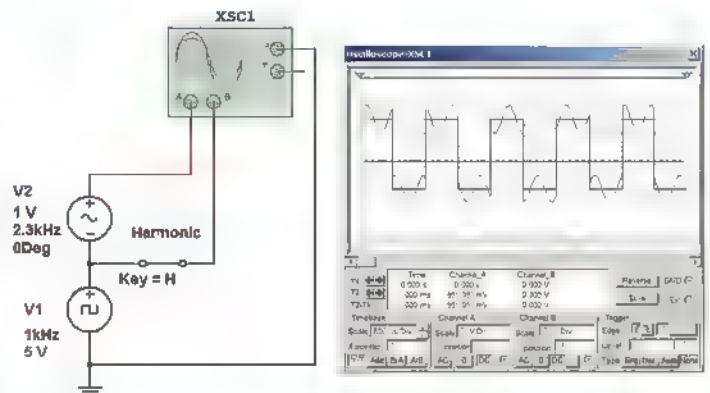
Using the circuit in **Figure 2**, we can shift the amplitude, phase, and frequency of the harmonic. There are some key points to be made here. Harmonics produce variations in the shape of the base wave. The higher the amplitude of the harmonic frequency, the greater the change in the base wave's shape. The higher the multiple the harmonic frequency is, the greater the number of changes that occur to each base wave cycle. These concepts are demonstrated in **Figure 4**.

In **Figure 5**, I took one of the signals in **Figure 4** and (using Microsoft Paint) shifted the position of the traces so as to recreate the harmonic signal. Below that, I compared one cycle to another by overlapping the signals.

Let's look a little more deeply at harmonics. The thing that is significant about harmonics is that they are stable,



**FIGURE 4.** The effect of harmonics on base signals.



**FIGURE 6.** Square wave with non-harmonic interference highlighted in red.

repeating every cycle. If I shift the harmonic in **Figure 2** by 300 Hz, you will see that the red signal shifts in phase between cycles of the square wave. **Figure 5** shows a square wave with a non-harmonic signal at near twice the base frequency.

Visualizing the frequency content of a waveform is challenging. This is most true with live signals with their constant small changes. If possible, store the trace and look back at it in a stationary form. Focus in on the characteristics of the waveform that are stable, keeping in mind how the base waveform looked. **Figure 6** shows the original base waveform placed on top of the harmonic signal to highlight the differences. You are looking for variations in the signal that repeat from cycle to cycle, not just random changes in the signal. **Figure 7** is an illustration of a square wave with non-harmonic interference. Note the variation in the change between cycles.

The differentiation between harmonic and non-harmonic interference is important. Harmonics are caused by non-linear changes in current and voltage. Harmonic interference is commonly the result of a distortion to the base signal, such as clipping with a diode. Non-harmonic interference is from a source unrelated to the base signal.

Next, let's try to identify specific harmonic frequencies. The closest we can get to this with our technique is to identify the harmonic number. The





The variation in the change between cycles shows that the interference is non-harmonic

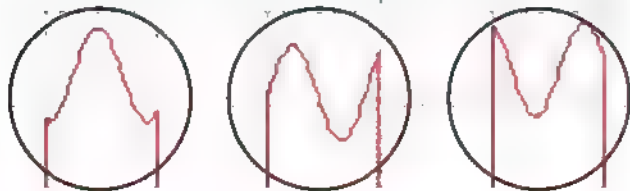


FIGURE 8. Interpreting non-harmonic interference.

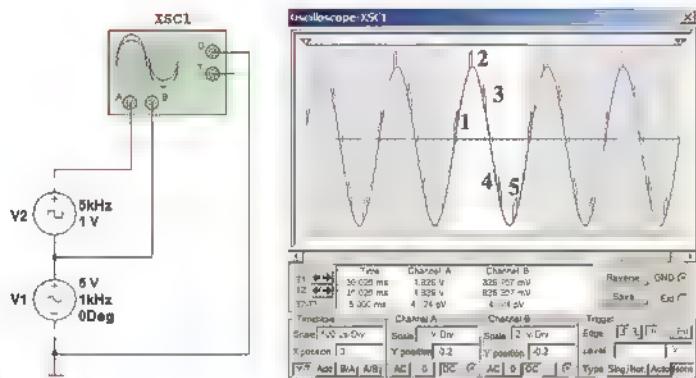


FIGURE 9. Sine wave with a square wave at the fifth harmonic.

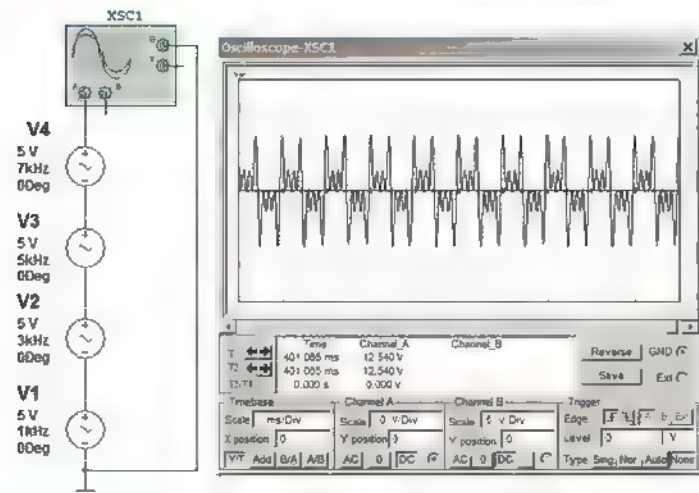


FIGURE 10. Signal produced by equal level odd harmonics.

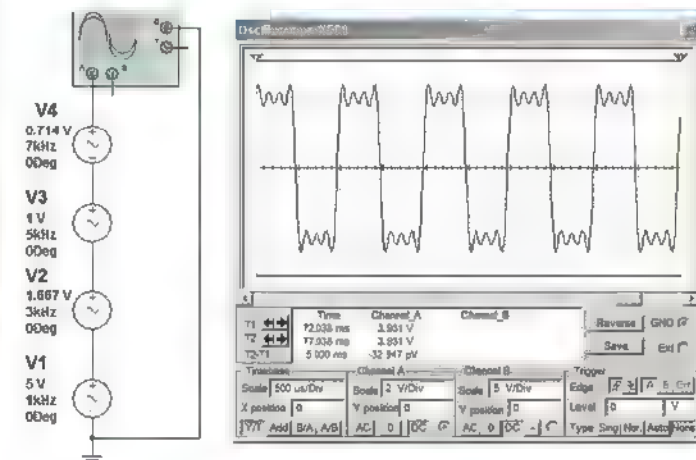


FIGURE 11. Waveform with harmonic levels adjusted to approximate a square wave.

harmonic number is the multiple of the base frequency that the harmonic frequency equals. For example, if the base frequency was 1,000 Hz, the third harmonic would be 3,000 Hz.

This technique is not good for harmonics alone. It can also be used to identify near-harmonic interference. This is in many cases as accurate as necessary, for example, when selecting a filter. Make note of the number of changes that occur within a waveform cycle and the shifting position of the changes. This is easiest when you're able to observe several cycles at once.

To help visualize this, I simulated a square wave harmonic on a sine wave base frequency. This caused the harmonic to produce a definitive red pulse on the base sine wave. In **Figure 9**, I generated a sine wave with a square wave set at the fifth harmonic. The number of the harmonic corresponds with the number of times the change occurs in a cycle. In this case, the change that occurred was five red pulses that appeared in each full cycle of the sine wave.

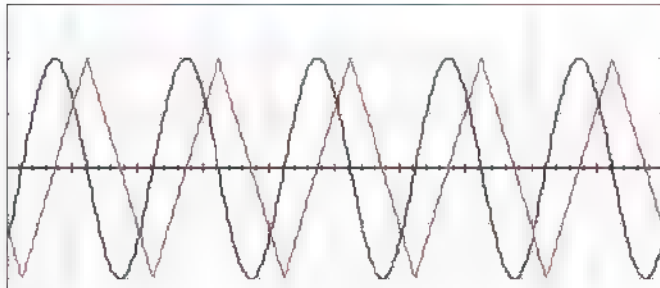
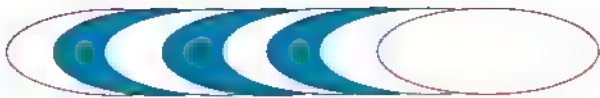
When I went to school, I was taught that a square wave consisted of a fundamental frequency and all its odd harmonics. If we virtually reverse-engineer a square wave and reassemble it from its harmonics, we will see that this is not entirely true. As shown in **Figure 10**, when

attempting to add a fundamental frequency and its odd harmonics, nothing approximating a square wave results because the harmonic formula is incorrect.

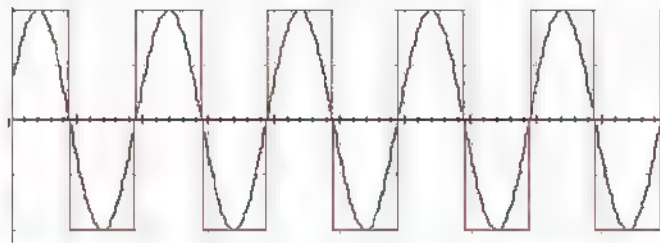
In a square wave, the level of each harmonic should equal the base amplitude divided by the harmonic number. Observe how **Figure 11** more closely approximates a square wave.

Both triangle and square waves contain the same harmonics; it is only the harmonic mix that is different. Triangle waves contain less energy in their harmonics, and as a result more closely resemble a sine wave (as illustrated in **Figure 12**). It is often not the frequency of a harmonic that matters, but the energy contained in the harmonics that could possibly result in interference. The magnitude of the change to a sine wave is an indication of the energy contained in the harmonics. By noting the differences between a signal and a sine wave, one can start to gauge the energy in the harmonics.

In a square wave, the amplitude of the harmonics is determined by dividing the amplitude of the base frequency by the harmonic number. In a triangle wave, the amplitude of the harmonics is determined by dividing the

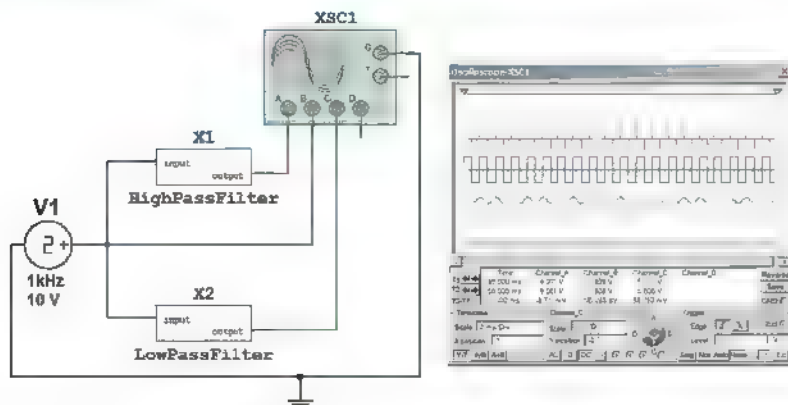


**A Triangle wave more closely resembles a Sine wave**



**than a Square wave**

**FIGURE 12. Triangle waves more closely resemble sine waves.**

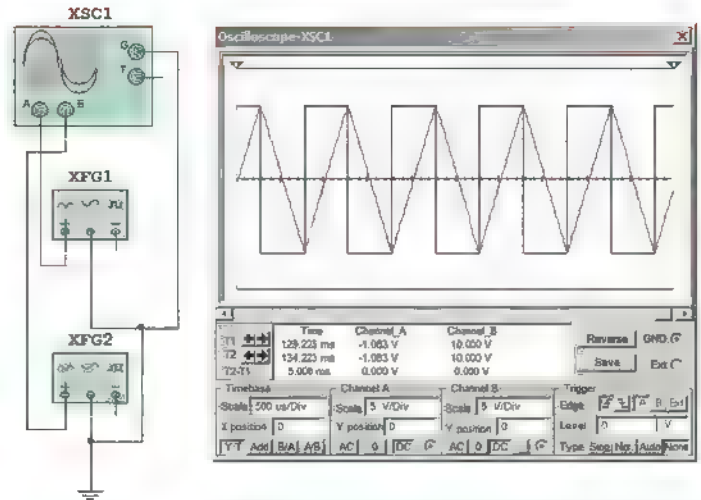


**FIGURE 14. Square wave filtering.**

amplitude of the frequency by the square of the harmonic number. **Figure 13** shows a dual trace of a square wave and triangle wave overlapping. The red square wave contains more energy in its harmonics. The rapid change of signal level and the flat regions in the square wave are indicative of higher energy levels in the harmonics.

This information can form a base line for testing filter circuits. In **Figure 14**, a square wave was sent through both a high pass and a low pass filter. The high pass filter blocked the base frequency while passing the higher harmonics; this produced spiking or rapid changes in levels which is indicative of high frequencies. The low pass filter passing the base frequency while blocking the higher harmonics produced a waveform resembling a sine or triangle wave, which is indicative of lower energy in the harmonics.

The comparison of the output of a known good filter



**FIGURE 13. Triangle and square waves.**

to the one under test can provide valuable information. If known, select a square wave frequency in the range of the filter circuit, or otherwise just adjust the square wave frequency until you see a significant change in the filter output.

When using a building block approach to produce a square wave, it is impossible to supply every harmonic. So, the question is how many harmonics are adequate. The answer is the more harmonics used, the faster the square wave will change levels, and the longer the flat areas will be on the top and bottom.

**Figure 15** shows a signal made of just the first and third harmonic in black compared to a real square in red. You can see the flat areas of the square wave are larger and the time it takes for the square wave to change levels is faster. The good square wave is indicated by the reduction in rise and fall times and the longer flat regions.

Square waves can be used to evaluate sound systems. In **Figure 16**, a square wave was input into a simulated speaker where the impact was captured on an oscilloscope.

This technique can be used to detect high and low frequency roll-off, as well as over-shoot and ringing. If you are heavily into sound engineering and want to use this process to evaluate systems, you will need a high bandwidth square wave in order to capture all the harmonic information. This combined with the FFT (Fast Fourier Transform) feature now available on many oscilloscopes offers a great tool for the comparison of sound systems.

## Final Thoughts

With simulation programs and a building block

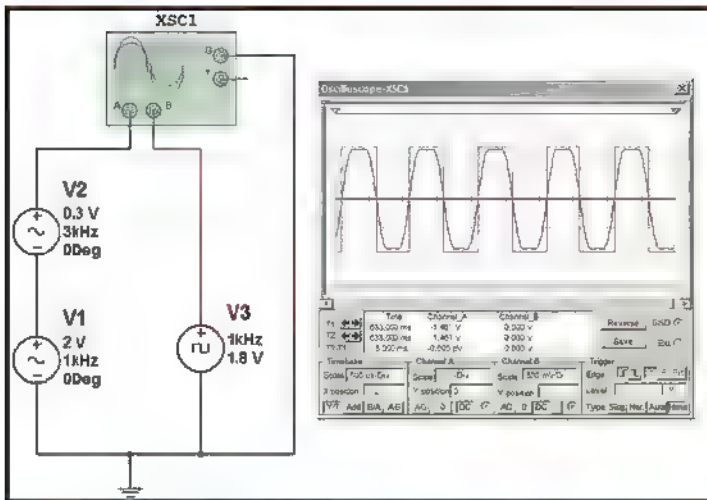


FIGURE 15. Limited harmonic simulated square wave.

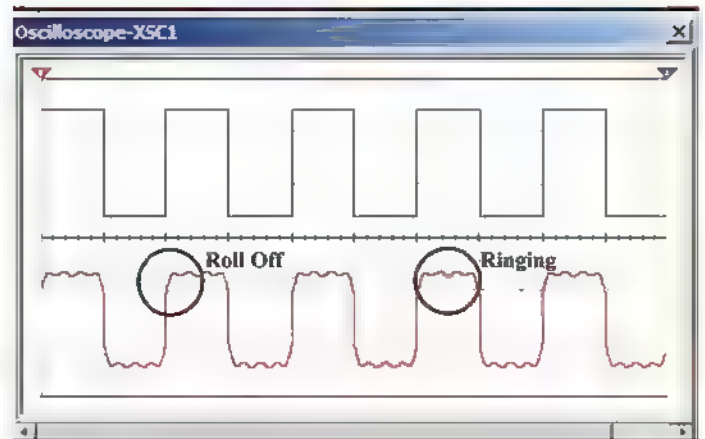


FIGURE 16. Roll off and ringing.

approach, you can systematically look at the structure of complex signals. The building block approach allows for the testing of electronic circuits under very controlled conditions. This technique also enables precise control of signal bandwidth and spectral content.

The building block approach can be used to produce

a variety of complex wave forms including AM and SSB. It can be used to inject noise into a signal and test the performance of various types of detectors.

These simulation techniques can be looked at by the technician and the technical educator as tools for testing, comparing, and explaining various circuits. **NV**

# 500 MHz

# \$5,000

MiniUnit Touch Screen Scope  
 100MHz @ 100ns  
 2.5 Turn Power



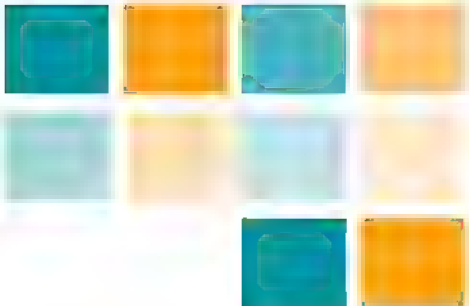
See it now at [teledynelecroy.com/wave5](http://teledynelecroy.com/wave5)

**TELEDYNE LECROY**  
Everywhere you look™

# Breaking the Arduino Speed Limit Part 2

By Bob Davis

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?/magazine/article/october2015\\_Davis](http://www.nutsvolts.com/index.php?/magazine/article/october2015_Davis).



In essence, this is a continuation of my previous article, “Breaking the Arduino Speed Limit” (March 2014) where I introduced the CA3306 fast analog to-digital converter (ADC) and a 128x64 LCD screen to display the results. That article created so much interest that I decided to write a second article. Here, I will show the solution to the clock “glitches,” add a serial LCD screen that leaves more Arduino pins available for other things, and introduce an eight-bit fast ADC. Then, we will put it all together to make an even nicer digital storage oscilloscope.

The glitch problem is caused by the clock for the ADC not matching the clock for the Arduino (**Figure 1**).

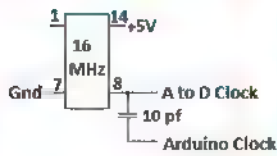
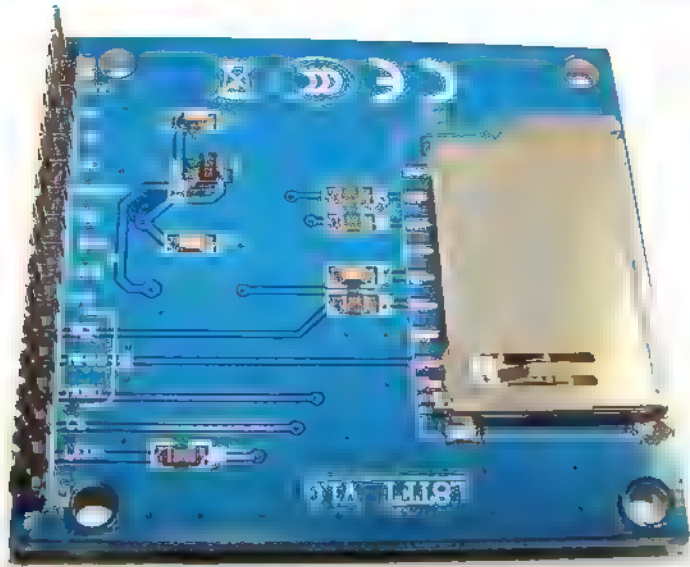
For a long time, I tried to tap into the Arduino clock on pin 9 and use that clock for the ADC. Then, I tried overriding the Arduino clock with an external clock, but that did not work either. Then, I found the solution. If you couple an external 16 MHz clock through a 10 or 20 pF capacitor to pin 9, it works! The problem is that pin 9 is used to having a very small signal on it, so the external clock has to be very small as well. This solution only works for the Arduino Uno as it is a hardware modification.

The improved clock synchronizing circuit is just a 16 MHz oscillator with a 10 pF capacitor that goes to the Arduino IC pin 9. You could also use a 32 MHz clock and divide it by two first. To be honest with you, I have also tried replacing the 16 MHz oscillator with a 24 MHz oscillator and the Arduino Uno clocked up to 24 MHz! Refer to the schematic in **Figure 2**.

Before we get started with the ADCs, let’s change out the LCD screen for one with color and a serial interface so it will use less Arduino I/O pins. A popular serial LCD is the 1.8TFT SPI 128x160. It only needs five I/O pins with

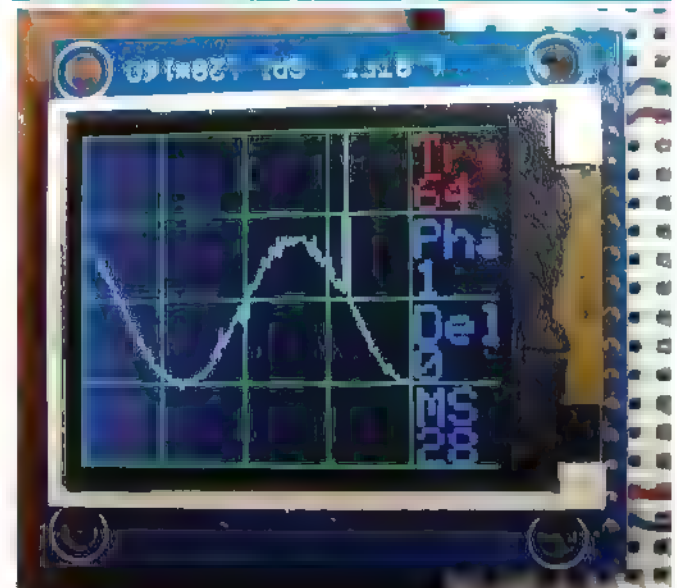


**FIGURE 3** Back side of 1.8 TFT LCD.



**FIGURE 2** Clock schematic.

**FIGURE 4** Glitch from lack of clock synchronization.



power and ground to operate.

There appear to be two versions of the 1.8 inch TFT LCD screen. One version has 10 pins and the other version has 16 pins. The pin definitions are written on the bottom of the circuit board, so it is just a matter of writing them down before you flip it over and wire it up. In **Figure 3**, you can see that the pins are clearly marked

The programs for this LCD will use the new built-in TFT drivers found in version 1.0.5 of the Arduino driver.

The programs will not work without this TFT driver being properly installed. **Table 1** is a chart showing how to wire the Arduino Uno to the LCD screen. I ran jumpers to connect the two grounds and 5V pins together on the breadboard.

Note that pin 1 of the LCD is on the right and pin 16 is on the left as you look at the top of the LCD screen. **Figure 4** is the schematic diagram of how to wire the LCD screen up. Note that D0-D7 and the six analog pins are

**Table 1.**

Arduino Uno	1.8 SPI TFT
GND	Pin 01 (GND)
5V (VCC)	Pin 02 (VCC)
Not used	Pin 03
Not used	Pin 04
Not used	Pin 05
D8	Pin 06 (RESET)
D9	Pin 07 (A0)
D11 (MOSI)	Pin 08 (SDA)
D13 (SCK)	Pin 09 (SCK)
D10 (SS)	Pin 10 (CS)
Not used	Pin 11 SD Card
Not used	Pin 12 SD Card
Not used	Pin 13 SD Card
Not used	Pin 14 SD Card
5V (VCC)	Pin 15 (LED+)
GND	Pin 16 (LED-)

**FIGURE 4** Arduino to-LCD schematic.

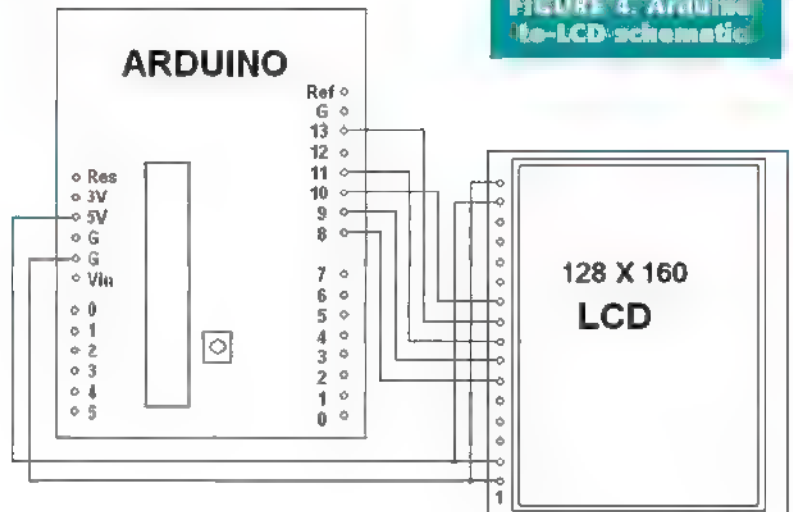


FIGURE 5. Simple Arduino oscilloscope.

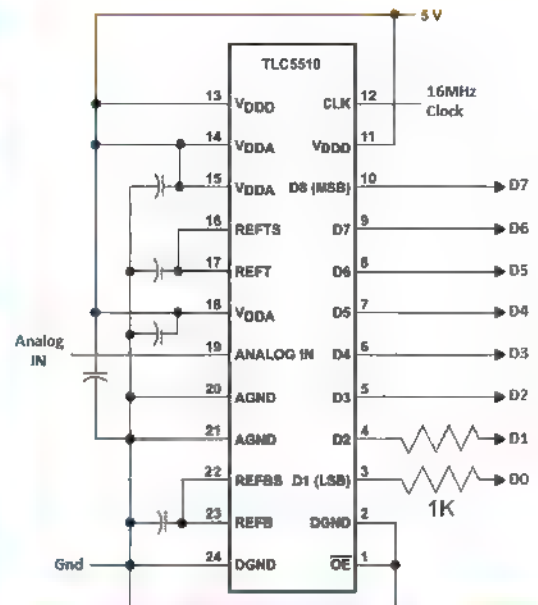
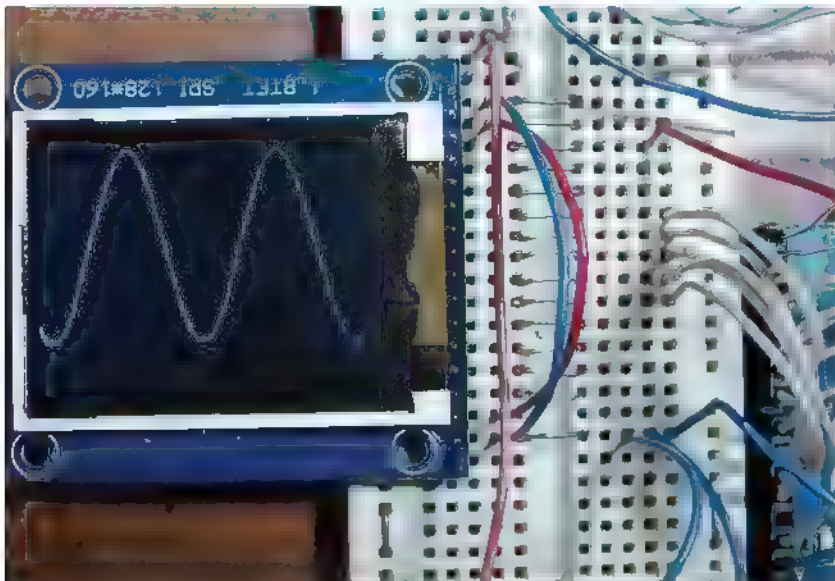


FIGURE 6. TLC5510 schematic.

now all free to be used for this project. We can start with a simple oscilloscope demonstration program that displays what is on A0 to test out the display. For a very simple analog input circuit, you can use a .1  $\mu\text{F}$  capacitor and a 10K variable resistor to provide bias. **Figure 5** shows a simple Arduino oscilloscope.

This sketch produces a simple oscilloscope that is

good to about 1 kHz. The oscilloscope trace is white:

```
// 1.8 SPI TFT Quick Oscope
// Reads & Displays the value of analog input
// on A0
// Created 15 January 2015 by Bob Davis
#include <TFT.h> // Arduino LCD library
#include <SPI.h>
// pin definitions for the Uno
#define rst 8
#define dc 9
#define cs 10
TFT TFTscreen TFT(cs, dc,
rst);
// set up variables
int Input 0;
byte Sample[160];
void setup()
// Initialize the display
TFTscreen.begin();
}
void loop(){
// Quickly collect the data
for (int xpos 0; xpos
<160; xpos++){
Sample[xpos]
analogRead(A0)/8;
}
// Erase the screen to start
// again
TFTscreen.background(0, 0,
0);
// select the color red and
// display data.
TFTscreen.stroke(0, 0, 250);
for (int xpos 0; xpos
<159; xpos++){
// draw the line (xPos1,
// yPos1, xPos2, yPos2);
TFTscreen.line(xpos,
Sample[xpos], xpos+1,
Sample[xpos+1]);
}
}
```

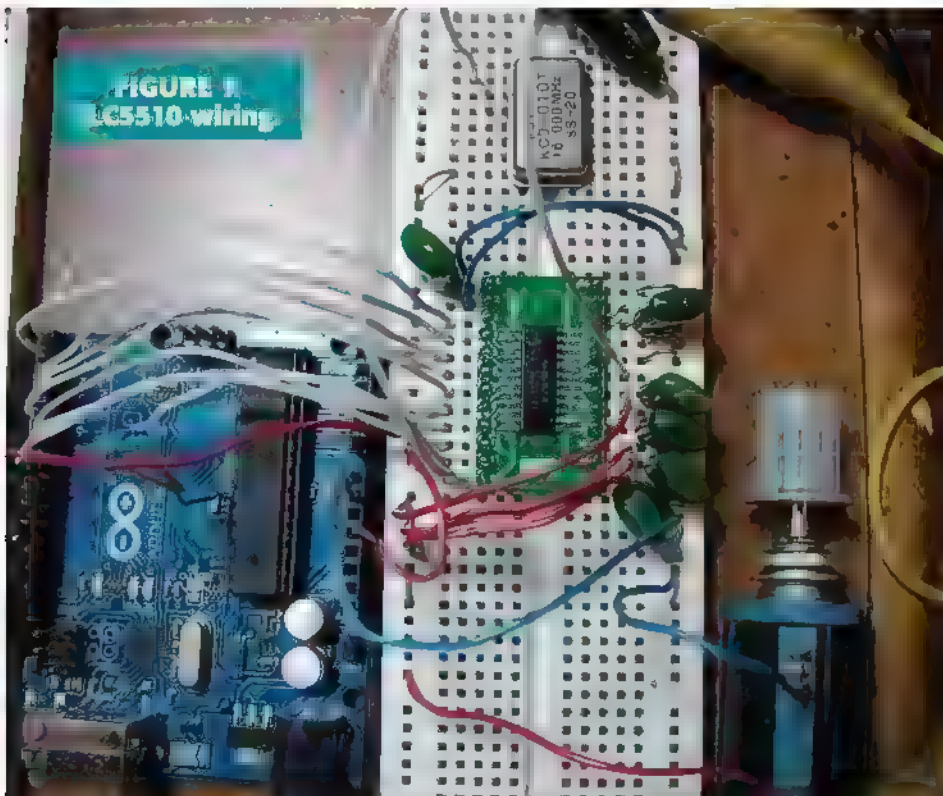




FIGURE 8. Pushbutton switches schematic.

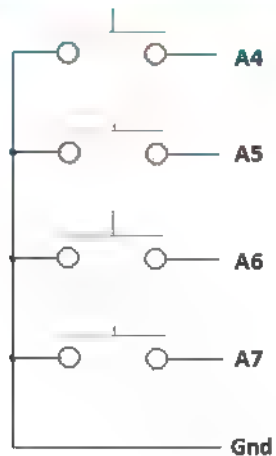
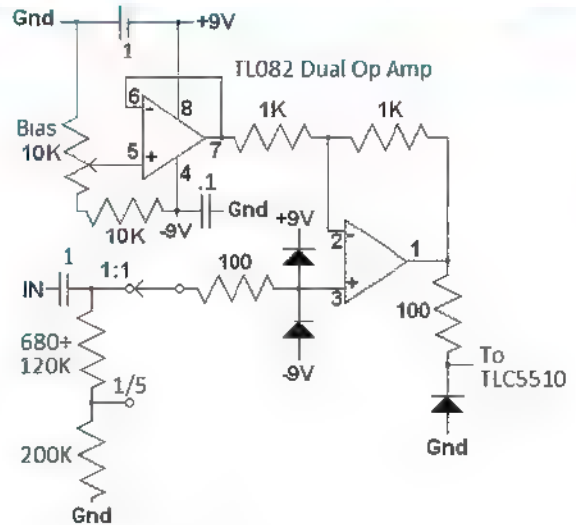


FIGURE 9. Analog input schematic.



Now, let's have some fun and add an ADC on D0 to D7. In software, the *PINC* parallel input command that was used in my previous article is replaced with *PIND*. We used the CA3306 – a six-bit/fast ADC – in the previous article, so now we will introduce the TLC5510. The TLC5510 is an eight-bit ADC. It comes in a SOP package, so you will have to use an adapter to plug it into a breadboard.

Note that the TLC5510 outputs now go to D0 to D7 of the Arduino. You will need to either disconnect D0 and D1 in order to update the software on the Arduino or use two 1K ohm resistors as shown in the schematic. These same D0 and D1 pins are always used for the Arduino to communicate over the USB connection. However, this is the only way to get eight bits in parallel into an Uno.

Here are some fast ADCs that are all somewhat "interchangeable:"

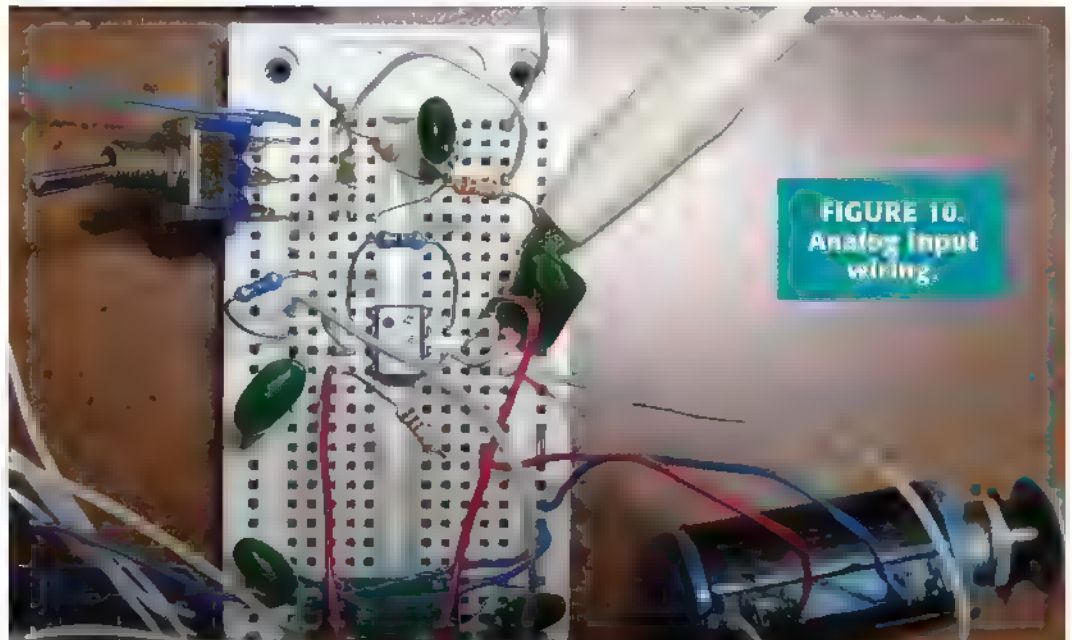
- The TLC5510 is eight-bit at 20 million samples per second.
- The TLC5540 is eight-bit at 40 million samples per second.
- The ADC1175 is eight-bit at 20 million samples per second.
- The ADC1175-50 is eight-bit at 50 million samples per second.

Figure 6 is the TLC5510 schematic

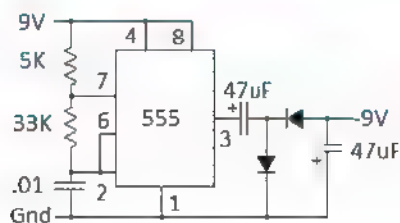
diagram. Note that pin 1 is at the bottom, so the IC is being drawn upside down. I wired it up upside down to match the schematic and then spun the breadboard around to hook it up. The capacitors are all .1  $\mu$ F at 16 volts.

Figure 7 shows the TLC5510 wired up and running. In this photo, you can also see the 10 pF capacitor going from the clock to pin 9 of the Arduino Uno.

Now to put some of those free analog input pins to work. You can connect four momentary contact switches to A2 to A5 to select up, down, right, and left. Up and down will select the item to change, and right and left will change the values of the selected item. You could put all four switches on one analog pin by having each one



**FIGURE 11: Negative power supply.**



select a different voltage. **Figure 8** shows the switches schematic.

Next, we will improve on the analog input section that we saw in the last article. In the previous design, when you changed the gain of the op-amp, the bias changed and had to be re-adjusted. It is better to not have an adjustable gain to avoid this problem. This change will even make the analog input section simpler to design. You can use a TL082 IC instead of the LF353 for the dual op-amp.

The “bias” control provides a bias to the ADC. You will need a bias of about 1.3 volts as 2.6 volts is the

maximum input. The 1.3 volts then becomes the “zero” position in the middle of the LCD screen. This bias is the 0 signal since the ADC uses 2.6 volts as its maximum input voltage and does not allow any negative input voltages. **Figure 9** shows the analog input schematic.

**Figure 10** is a photo of what the analog section looks like. Do not forget the .1  $\mu$ F filter capacitors near pins 4 and 8 of the op-amp. These capacitors will make a huge difference in the amount of noise. I left out the protection diodes for now, but they will need to be there for real world situations.

For a -9 volt source, you can use a 9V battery or (even better) use a 555 oscillator with two diodes to form a simple power inverter. **Figure 11** is a schematic for a 555 inverter. The diodes can be a 1N4001 or similar. Under load, the output is only about -5 volts, but that works fine.

Last of all, there are many software improvements. We can now switch between multiple selections like the delay between samples (as before), the trigger level, and the trigger phase.

Another software improvement is that the trigger is now very “solid.” It first checks for a positive-going signal, then for a negative-going one. Otherwise, it might just

The Easiest Way to Design Custom  
**Front Panels & Enclosures**

Free Front Panel Designer

You design it to your specifications using our FREE CAD software, Front Panel Designer

We machine it and ship to you a professionally finished product no minimum quantity required

- Cost effective prototypes and production runs with no setup charges
- Powder-coated and anodized finishes in various colors
- Select from aluminum, acrylic or provide your own material
- Standard lead time in 5 days or express manufacturing in 3 or 1 days

**FRONT PANEL EXPRESS**

FrontPanelExpress.com

**WORLD'S MOST VERSATILE**  
**CIRCUIT BOARD HOLDERS**

Our Circuit Board Holders add versatility & precision to your DIY electronics project. Solder assemble & organize with ease.

VISIT US ON

**MONTHLY CONTEST**  
Visit us on Facebook® to post a photo of your creative PanaVise project for a chance to win a PanaVise prize package

Model 201

**PANAVISE®**  
Innovative Holding Solutions

7540 Colbert Drive • Reno • Nevada 89511 | 800 759 7535 | www.PanaVise.com

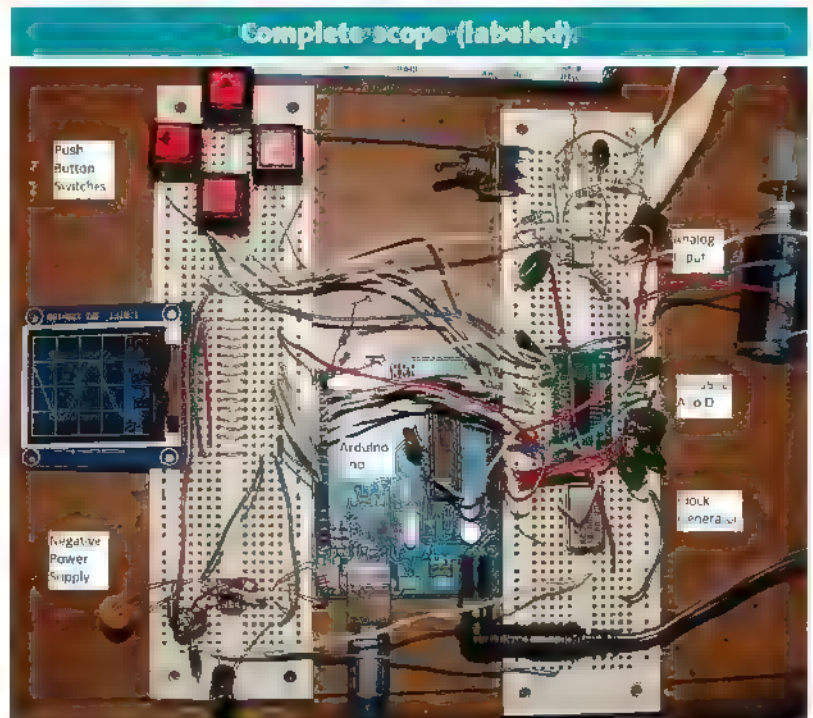


catch the positive level anywhere and proceed to collect samples:

```
// wait for a positive going trigger
if (trigphase 1){
  while (Input < trigger){
    Input PIND; }
  while (Input > trigger){
    Input PIND; }
}
```

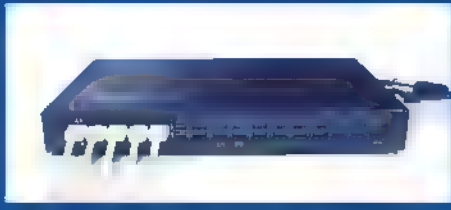
The future possibilities are endless. There are three pins still available on the Arduino: D12, A0, and A1. They could be used for logic analyzer inputs, or for additional analog inputs of just about anything that you can imagine. The free pins can even be used to control a FIFO memory IC that can increase the sample rate to 50 MSPS (Million Samples per Second) or even faster.

**NV**



High Quality Professional Instruments  
New F-series LA

**LAP-F1 Logic Analyzer**  
First choice for high-speed, comprehensive measurements



- Sample rate (Timing mode): up to 1 GHz
- Acquisition channels: 40 or 64
- Memory per channel: 4Mb, 8Mb, 16Mb, 32Mb or 64Mb
- 6 protocol triggers (hardware): I2C, I2S, SPI, SVID, UART, CAN 2.0
- eMMC 5.1 / SD 3.0 LA mode, protocol decoder and trigger
- More than 100 built-in protocol decoders
- Long-time records: Transfer via USB 3.0 to PC to view or sample
- Channel Folding: Disable channels to concentrate memory on the active ones
- OSO connection

**Active Probe features**



- Included in the LA purchase
- Good impedance matching, reduced crosstalk and noise and reinforced ground enhance the measurement quality, accuracy and stability of high-speed signals
- Support DUT bandwidths of up to 200 MHz
- 4 types: Standard, Low voltage, Negative logic and eMMC 5.1/SD 3.0 support



**Distributor Wanted**  
Tel:+886 2-66202225 #221 or #311

[www.zeroplus.com.tw](http://www.zeroplus.com.tw)

## *A Tiny, Wi-Fi Enabled, Arduino Compatible Microcontroller*

# Meet the ESP8266

*It is not very often that a new piece of hardware comes along and immediately captures the attention of the entire maker community. The Raspberry Pi and the \$9 C.H.I.P. are a couple of recent examples, but the ESP8266 module from Expressif Systems ([expressif.com](http://expressif.com)) wins this prize. This little board (see **Photo 1**) is only about the size of a nickel, yet contains a powerful 32-bit microcontroller and a Wi-Fi interface, and it can be purchased for around \$4 in single unit quantities.*

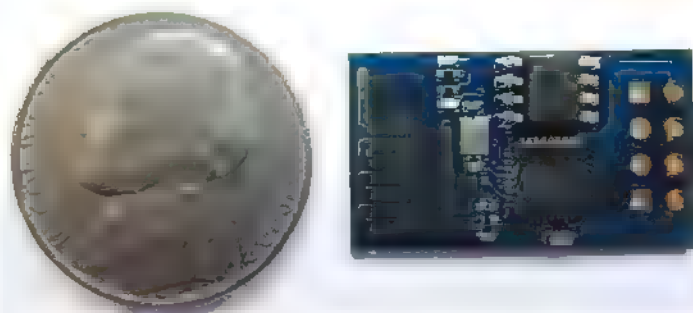
**T**he first projects built with this module all used a microcontroller to control the ESP8266 as a Wi-Fi peripheral using an AT command set over a serial interface. While this was made to work, some of the projects suffered from stability problems as the ESP8266 firmware continued to evolve. Lately, however, a group of enterprising individuals have made the ESP8266 Arduino compatible. This is important for numerous reasons:

1. It allows people familiar with the Arduino IDE (Integrated Development Environment) to develop software for the ESP8266 module.

By Craig A. Lindley

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php7/magazine/article/october2015\\_Lindley](http://www.nutsvolts.com/index.php7/magazine/article/october2015_Lindley).

**Photo 1.** The ESP8266 module (ESP-01) is about the size of a nickel



2. It allows the software developed in the Arduino IDE to be run directly on the 32-bit microcontroller on the ESP8266 module, eliminating the need (in many cases) for a separate microcontroller altogether.

3. It allows the use of numerous third-party Arduino libraries, as long as they don't depend on direct access to the underlying AVR hardware.

Arduino compatibility and the low cost of the ESP8266 are major developments for the Internet of Things (IoT) movement currently sweeping the tech world. Using the ESP8266 allows for very small and inexpensive products to be created that can be controlled and/or monitored remotely.

Note if you plan on putting an ESP8266 module into a commercial product, you will have to pass FCC

**Photo 2.** ESP8266 (ESP-01) pinout. The squiggly trace is the Wi-Fi antenna. This device has 512K of Flash for program storage



certification, which can take considerable time and be rather costly.

To understand what a breakthrough this is, consider the cost and size of a traditional Arduino approach to Wi-Fi enabled monitoring and control. First, you have to have an Arduino board (say, an Arduino Uno) from a reputable source which costs between \$20 - \$30. Then, you have to purchase a Wi-Fi shield for around \$20 - \$40, bringing the basic system cost to between \$40 - \$70. Then, consider size. The Uno's dimensions are 2.1" x 2.7". Attach the Wi-Fi shield and the sandwich is between 1.25" to 1.75" deep and a bit harder to package than the ESP8266 (again, which is the size of a nickel).

Finally, when you consider the ESP8266 has a 32 bit processor which can run at 160 MHz — 10x the speed of the Uno's eight-bit processor — and that it has 512K (minimum) of Flash memory program space to the Uno's 32K, the Uno Wi-Fi solution is looking a little dated.

## The Hardware

Actually, the ESP8266 is a whole family of modules which vary in the number of available I/O pins, the amount of onboard memory, the types of interfaces available, and in how the RF antenna is attached/implemented. The module I will be describing in this article (and shown in **Photos 1** and **2**) is referred to as an ESP-01. This module has its RF antenna etched directly onto the circuit board.

Information on the whole family of ESP8266 devices is available at [www.esp8266.com/wiki/doku.php?id=esp8266-module-family](http://www.esp8266.com/wiki/doku.php?id=esp8266-module-family).

The following attributes of the ESP8266 family were extracted from the datasheet (see [https://nurdspace.nl/File:ESP8266\\_Specifications\\_English.pdf](https://nurdspace.nl/File:ESP8266_Specifications_English.pdf)):

- 802.11 b / g / n

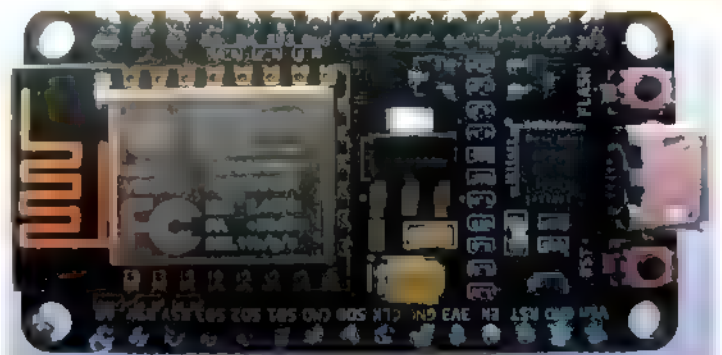
- Wi Fi Direct (P2P), soft-AP
- Built-in TCP / IP protocol stack
- 802.11b mode + 19.5 dBm output power
- Built-in temperature sensor
- Supports antenna diversity
- Off leakage current is less than 10  $\mu$ A
- Built-in low power 32-bit CPU which can double as an application processor
- SDIO 2.0, SPI, UART, ADC
- Standby power consumption of less than 1.0 mW (DTIM3)

In other words, the ESP8266 family of modules features low power consumption, high RF power output, and is capable of supporting all of the current 802.11 standards required for Wi-Fi connectivity. In addition, it supports many industry standard hardware interfaces and can also function as the application processor in many designs. Note: The ESP8266 is a 3.3 VDC part.

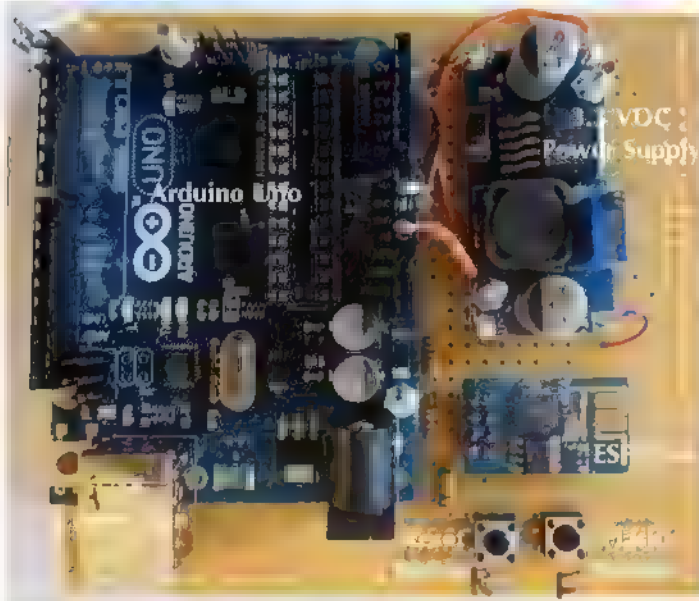
## Prototyping Hardware

Now that we know something about the ESP8266 module family, let's talk about what we need in terms of hardware to try out the ESP8266 in the Arduino environment. In addition to the ESP8266 ESP-01 module, we need some sort of USB to-TTL serial adapter, a 3.3 VDC power supply capable of at least 250 mA of output current, a couple of momentary pushbutton switches, an LED, and a 1K and a 10K ohm resistor. Do not skimp on the power supply for the ESP8266. It requires quite a bit of current, and lack of sufficient current will cause the ESP8266 to appear flakey or not work at all.

**Photo 3.** An ESP8266 (ESP-12) development module called the NodeMCU Amica with many more I/O pins and memory (4 MBytes) is available. It functions just like the prototype hardware described in this article and is powered directly from the USB port.



**Photo 4.** Prototyping hardware using an Arduino Uno. This method is not recommended, but works



When I received my ESP8266 modules in the mail, I was anxious to try them out but I didn't yet have the required USB-to-TTL 3.3V serial interface cable to proceed. Since necessity is the mother of invention and since I am not known for being a patient person, I decided to use an Arduino Uno board I had for this purpose. Note, I removed the processor from the Uno as it was not needed.

The breadboard is shown in **Photo 4** and the Fritzing schematic is in **Figure 1**. Here, the 5V output of the Uno was used to drive a small switching mode power supply set to 3.3 VDC which, in turn, drives the ESP8266. The Tx and Rx signal from the Uno were connected directly to the Tx and Rx connections on the ESP8266. Yes, Tx to Tx and Rx to Rx. Remember, the ESP8266 is a 3.3V part and that direct connection to the 5V logic levels of the Uno

should be avoided. With that being said, this prototype worked perfectly. I have since heard the ESP8266 has 5V tolerant pins but I have yet to have that claim substantiated. Anyway, I figured I only had a few bucks tied up in the ESP8266 part so if it blew, oh well. As it turned out, this prototype worked splendidly.

The Reset pushbutton on the prototype pulls the reset pin on the ESP8266 low, thereby resetting the device. The Flash pushbutton grounds the GPIO0 pin which places the ESP8266 into firmware download mode. The *CH\_PD* line must be pulled high for new firmware to be downloaded.

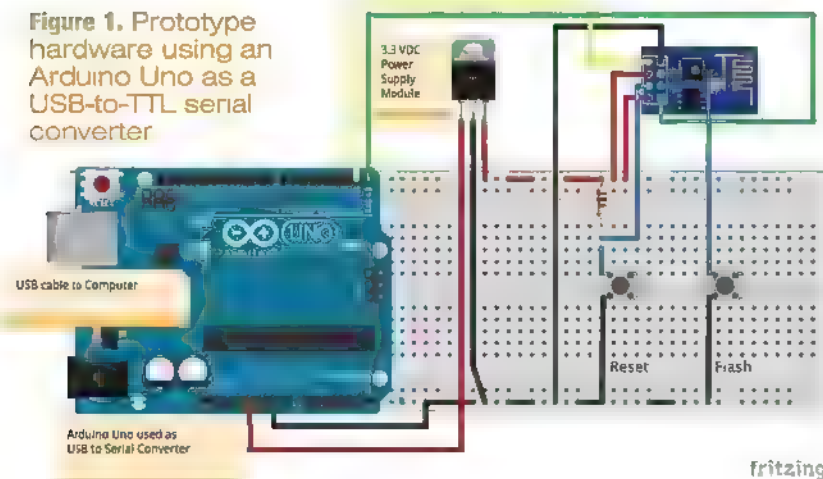
After my 3.3V USB-to-TTL serial cable arrived, I removed the Uno from the prototype and connected the cable directly to the ESP8266. This approach is shown in **Photo 5** and the schematic in **Figure 2**. Here, the cable provides 5 VDC on the VCC pin which is connected to the 3.3 VDC power supply. The Tx and Rx pins of the cable are at the proper 3.3V interface levels. The Tx pin of the cable is connected to the Rx pin of the ESP8266, and the Rx pin of the cable connects to the Tx pin of the ESP8266. I also added an LED and 1K resistor connected between ground and GPIO2 which will be used with the Teleduino demo described later.

In either case, the following series of steps must be followed to initiate successful loading of code from the Arduino IDE into the ESP8266 module:

1. Press and hold the Reset button down.
2. While holding the Reset button down, press and hold the Flash button.
3. Release the Reset button while still holding the Flash button down.
4. Click the Upload button in the Arduino IDE.
5. When the sketch starts to load, you can release the Flash button.

Once code is successfully uploaded to the ESP8266, it will be executed every time a power-up or a reset occurs.

**Figure 1.** Prototype hardware using an Arduino Uno as a USB-to-TTL serial converter



## Arduino IDE Version 1.6.4

To easily program the ESP8266 as an Arduino, you must use the latest version of the Arduino IDE. As of this writing, that is version 1.6.4. This version has a feature called the board manager which lets third-party vendors add support for their Arduino compatibles that the makers of the IDE don't support directly. Adding support for the ESP8266 is a multi-step process:

1. First, you must download version 1.6.4 or newer of the IDE from [www.arduino.cc/en/](http://www.arduino.cc/en/)

**Main/Software** and install it. There are versions available for Windows, Mac OS X, and Linux. The installation process is different for each platform, but in each case is pretty straightforward. Follow the directions provided within the installation programs and you should be good to go.

2. Next, bring up the Preferences page of the IDE and type [http://arduino.esp8266.com/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/package_esp8266com_index.json) into the Additional Boards Manager URLs field. With this completed, click OK.

3. Next, go to the Tools menu tab in the IDE, click Board, and then Boards Manager. This will bring up a list of installable items. Scroll down and you should see esp8266 by the ESP8266 Community. Highlight this entry and an Install button should appear. Click this button to install the ESP8266 development software. This can take awhile because a lot of software is being transferred to your computer.

Once this process has been completed, the next time you click the Tools menu and then the Board entry you should be able to select the "Generic ESP8266 Module." You are now ready to program your ESP8266 as an Arduino. Be sure you make this selection for all of your projects which utilize the ESP8266 module.

In addition, make sure you have a serial port selected in the IDE so the serial monitor can be used. If an appropriate serial port does not show up in the IDE, you may have to install a driver for the USB serial cable you are trying to use.

## The Software

With either variety of prototyping hardware in place and the Arduino IDE updated with ESP8266 support, you are now ready to go. Right out of the box you have access to numerous example programs which illustrate some of the ESP8266 module's capabilities. If you go to File and then Examples in the IDE, you will see these three categories of example programs:

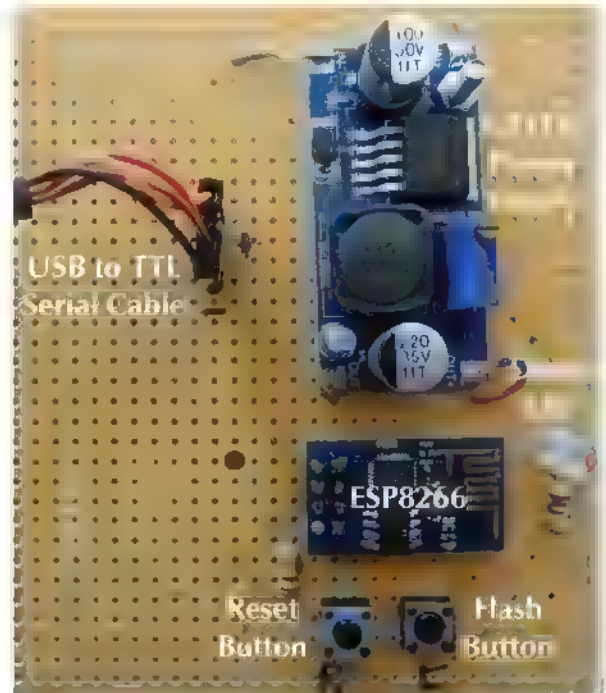
ESP8266mDNS  
 ESP8266WebServer  
 ESP8266WiFi

Each of these have one or more example sketches within them. In the ESP8266WiFi category, the following sketches are of special interest:

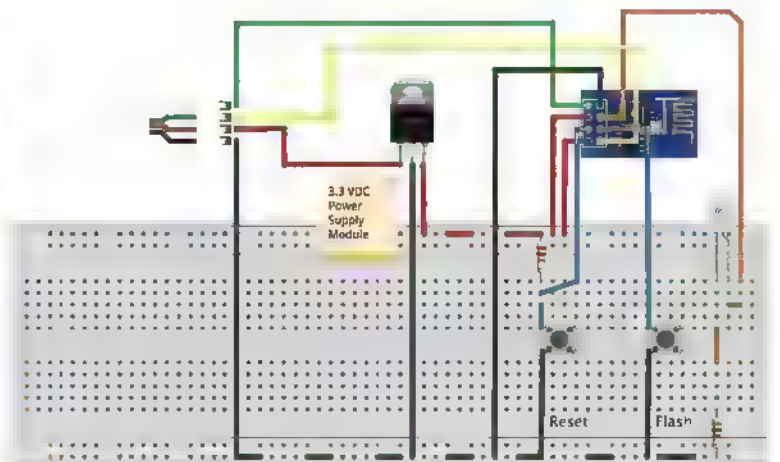


**Figure 2.** Prototype hardware using a USB to 3.3 VDC TTL serial cable. Includes an LED for the Teleduino demo.

**Photo 5.** Recommended prototyping hardware using a USB 3.3V TTL serial cable. The LED is used for the Teleduino demo.



1. NTPClient – shows how to get the time from a Network Time Protocol (NTP) server while demonstrating the use of UDP packets. This is the same technique your computer uses to set its time automatically.
2. WifiClient – shows how to use the ESP8266 to talk across the Internet (as a client program) to a server. All client applications will resemble this example program including the Teleduino client discussed shortly.
3. WifiScan – lists all of the wireless networks within range of the ESP8266. The network's name, signal strength, and whether or not the network is encrypted is



fritzing

displayed on the serial monitor. The list of networks is updated every five seconds.

4. WiFiWebServer — shows how to use the ESP8266 as an HTTP type server. By locally accessing this web server with a browser, an LED connected to the ESP8266 can be toggled off and on. See the example sketch for more information.

Many of these example programs/sketches must be edited before running as you must enter the SSID of your wireless network along with your network's password. Without this information, the ESP8266 will not be able to connect to your wireless network and the example programs will fail.

You may be wondering how much of the Arduino software environment has been ported to the ESP8266 and, in truth, the answer is quite a lot. A list of what is working is available at <https://github.com/esp8266/Arduino>. This list is definitive as these are the people who did/are doing the Arduino port.

## Teleduino

Using the ESP8266 to control an LED or some other

device on your local area wireless network is cool but somewhat limiting. What if you want to control your device from anywhere in the world instead? There are multiple ways of doing this — some of which require configuring your modem/router to forward messages through your firewall, opening up the possibility of security breaches. Nathan Kennedy of **KennedyTechnology.com** has come up with a better idea that he calls Teleduino.

Teleduino is designed primarily for use with an Arduino with a wired Ethernet shield; there are versions available for Uno and Mega based boards. I was interested to see if I could port some of the Teleduino functionality onto the ESP8266 as an experiment, and with Nathan's help I did so. Go to [www.teleduino.org](http://www.teleduino.org) for the details. If you are interested in the full Teleduino functionality on the ESP8266, you will have to wait for Nathan to port the complete code base. If, however, you want to experiment yourself, you can grab my code (TeleduinoClient.ino) from the article link.

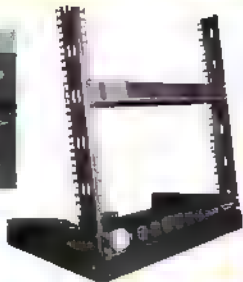
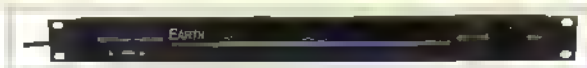
To use the Teleduino code, you must first go to [www.teleduino.org/tools/request-key](http://www.teleduino.org/tools/request-key) and request an API key. A key will be emailed to you after you provide your name and email address. A key is a long string of hex

**EARTH.LCD.COM**  
"The Smart LCD Company"

## Pi-RAQ

*An open source Raspberry Pi Based 1U Rack Mount Internet Appliance With the World's Only 10" x 1"*

With the innovation of the Pi-Raq and existing EarthLCD 10x1 display, 1U Rack mount equipment, such as power controllers, audio/video switchers, network hubs and routers, can now have an easily programmable front panel with a color TFT LCD



### Features at a glance:

- 10" x 1" Color TFT
- 1024 x 100 pixels
- 250 Avts
- 7 Function Tact and Scroll Wheel
- 20V Input Power Supply
- 10x1 LCD Adapter Board
- **Raspberry Pi 2**
  - 4 USB Ports
  - 100 MHz Ethernet Port
  - 900 MHz Arm Quad Core CPU
  - 1 GB RAM / 4 GB Micro SD Flash

Not exactly what you need? No problem! EarthLCD can customize a solution just for you!

### Contact Us:

For more information or to request a quote  
949.248.2333 Direct  
sales@earthlcd.com  
[www.Pi-RAQ.com](http://www.Pi-RAQ.com)

All logos and trade marks are respective property of their holder © Copyright 2015 Earth Computer Technologies, Inc.



## AP CIRCUITS

PCB Fabrication Since 1984

As low as...

# \$9.95

each!

Two Boards  
Two Layers  
Two Masks  
One Legend

**Unmasked boards ship next day!**

[www.apcircuits.com](http://www.apcircuits.com)



characters which must be edited into the TeleduinoClient sketch along with your Wi-Fi network's SSID and password before downloading into the ESP8266. The API key must also be used in all API calls from your browser.

Once the sketch is downloaded into the ESP8266, it will first make a connection to your Wi-Fi network and then it will open a TCP connection to the Teleduino server. Once this connection is made, the ESP8266 will authenticate itself to the server by providing the API key. If you bring up the serial monitor, you will be able to watch this interaction take place.

If all goes well, the Teleduino server will begin sending ping messages to the ESP8266 about every five seconds. Because the ESP8266 establishes an outgoing connection to the Teleduino server, there is no need to open any ports in your firewall and thus create any new security concerns for your network.

To summarize: Once the TeleduinoClient is configured, it automatically connects itself to the Teleduino server when powered up. The Teleduino server translates instructions received over the Internet into actions on the Teleduino device which (in this case) is the ESP8266.

As mentioned, I ported only a (very small) subset of Teleduino functionality. In fact, the TeleduinoClient sketch only recognizes a `setDigitalOutput` API call, but that is enough to prove the concept workable. On the second prototype shown in **Photo 5**, I have connected an LED to the ESP8266's GPIO2 pin through a 1K ohm resistor to ground. If I go to my browser and type in (this rather long URL) <https://us01.proxy.teleduino.org/api/1.0/328.php?k=<YOUR KEY GOES HERE>>

`&r=setDigitalOutput&pin=2&output=1&expire_time=0&save=0`, the LED will turn on. If I try this again but change `output=0`, the LED will turn off.

Keep in mind that for this demo, the ESP8266 is turning an LED off and on by way of commands entered into a browser which can be located anywhere in the world. If instead of an LED connected to the ESP8266 you connected a solid-state relay (SSR), you could control devices such as a light, a heater, an alarm system, etc. The possibilities are endless.

Want to control your Teleduino device from your Android smartphone or tablet? Check out apps like Teleduino Controller Pro V2 in the Google Play store.

## Resources

The following links provide further information about ESP8266 devices:

Information about the ESP8266 Arduino port can be found at <https://github.com/esp8266/Arduino>.

An ESP8266 forum full of useful information can be found at [www.esp8266.com](http://www.esp8266.com). The Expressif forum is available at <http://bbs.espressif.com>.

The Teleduino client sketch (TeleduinoClient.ino) is available at the article link.

## Conclusion

This article doesn't begin to describe the cool things that can be done using the ESP8266. I hope after reading this article you will come up with many ideas for your own projects. I have a few in mind that I may share in future articles if there is enough interest. Please let *Nuts & Volts* know if you would like other articles about using the ESP8266.

Devices like the ESP8266 make possible the idea of connecting almost anything to the Internet and controlling and/or monitoring them from anywhere in the world. The ESP8266 is a giant step forward in the Internet of Things (IoT) revolution. **NV**

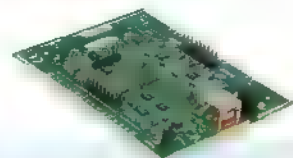
# Big Breakthrough in DAQ AT A SMALL PRICE

Easy to Use • Easy to Integrate • Easy to Support



## USB-201 Only \$99

- 8 analog inputs
- $\pm 10$  V input range
- 12-bit resolution
- 100 kS/s sample rate
- 8 digital I/O
- One 32-bit counter
- Support for Windows®, Android™, and Linux®



OEM board-only version is also available.

[mccdaq.com/USB201](http://mccdaq.com/USB201)



MEASUREMENT COMPUTING

Contact us

1.800.234.4232

©2015 Measurement Computing Corporation  
info@mccdaq.com

# Silent Sensors

By Joe T. Evans, Jr. and Spencer T. Smith

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?magazine/article/october2015\\_Evans](http://www.nutsvolts.com/index.php?magazine/article/october2015_Evans).

*We have all seen it so many times. Our hero returns to his (or her) room to find a hair fallen from the door sill. Out comes the Walther PPK! That hair, placed on the door sill before breakfast at a sunny sidewalk café, saves our hero. It functioned as a physical memory of the status of the door. The hair detected the occurrence or non-occurrence of a specific event and stored that information for retrieval by the spy returning to the room. This hair-based non-volatile memory had only two possible states that could be instantly read by the controller (our hero): in position (no event occurred) or not in position (the event occurred). Programming the hair meant placing it at a specific position on the door sill to be held in place by the door. If the door was opened during the retention period (breakfast), gravity let the hair fall from its pre-set position. No battery was needed for this event detector!*

## More Ferroelectric Magic

Ferroelectric capacitors make it possible to create an electronic version of the hair-based non-volatile event detector. Unlike more delicate forms of non-volatile memory that depend on static charges buried deep inside silicon dioxide, ferroelectric capacitors can operate in the natural world without being destroyed by Mother Nature's penchant for lightning large and small.

The February 2015 issue of *Nuts & Volts* featured an extensive article describing the physics of ferroelectric capacitors used as memory elements. Demonstrated in the article was a method of building and operating a simple non-volatile memory using a ferroelectric capacitor, a microprocessor, and a linear sense capacitor. In *this* article, we will show how to add a few more external components to that same memory bit to allow it to detect events and report them later.

## Review of Ferroelectric Memory

Following is a quick review of ferroelectricity. For a more complete description, please re-read the original February 2015 article (available at the article link). The crystal lattice of a ferroelectric capacitor is *naturally* polarized in the same way that a permanent magnet is permanently magnetized. This internal *electric* polarization cannot be detected with a voltmeter because the capacitor plates will always collect enough of the opposite charges to exactly cancel the polarization.

The internal polarization can be forced to change its direction by moving the cancellation



charge from one capacitor plate to the opposite plate. We read the state of the capacitor by monitoring the current flow that erupts from the capacitor when a reference voltage is applied to the capacitor. The stored datum corresponds to the *direction* of that internal polarization.

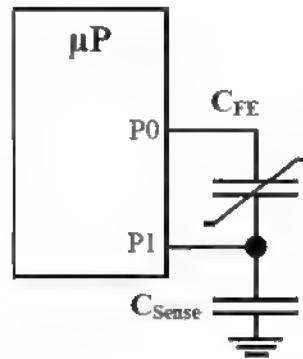
The plot in **Figure 1** shows two *single-cycle* hysteresis loops in terms of absolute charge for a ferroelectric capacitor starting from opposite dipole orientations. The horizontal axis is the voltage applied across the ferroelectric capacitor. The vertical axis is the count of the charges that come from the capacitor during actuation. A linear capacitor is also plotted for reference.

How much charge comes out of the capacitor when a positive voltage is applied tells us in which direction the ferroelectric capacitor pointed prior to the read operation. A linear capacitor in series with a ferroelectric capacitor can sense and report the amount of charge moving in

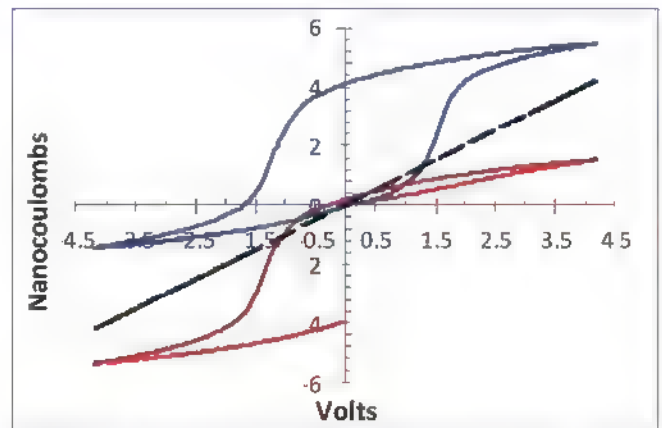
**Figure 1.** We attach this capacitor stack to two I/O pins of a microprocessor to create a single bit of non-volatile memory.

The linear capacitor must be sized so if the ferroelectric capacitor does not switch during a read operation, the small amount of charge that comes out of the ferroelectric capacitor into the linear capacitor cannot generate enough voltage on the common I/O pin to be read as a 1. If the ferroelectric capacitor does switch, so much more charge goes into the linear capacitor that its voltage on the I/O pin is read as a 1 by the microprocessor.

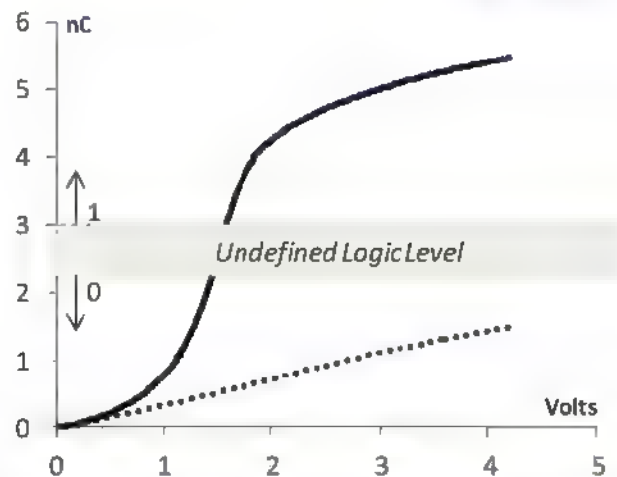
**Figure 2** redraws the ferroelectric hysteresis loops of **Figure 1** in terms of charge delivered at the top of the read voltage application.



**FIGURE 2.** Microprocessor-based non-volatile ferroelectric memory and a plot of its UP/DOWN memory states during a read operation.



**FIGURE 1.** UP and DOWN state full-cycle ferroelectric hysteresis loops compared to an equivalent-sized linear capacitor.



**Figure 2.)** Any sensor or generator that can source that energy can fully switch that ferroelectric capacitor. Attaching such a sensor to the ferroelectric capacitor in **Figure 2** creates an event detector. The microprocessor presets the ferroelectric capacitor in one direction and turns itself off. The sensor – if activated – moves the ferroelectric capacitor to the opposite direction. Upon waking up, the microprocessor executes a *read* operation. Was the ferroelectric bit in its original state or did it change?

There are a variety of readily-available sensors that can set the state of the Type AD capacitor. A solar cell can detect the presence of light in a room. A toy-scale electric motor will generate energy when its axle is turned by rolling motion. An antenna loop tuned to 13.56 MHz can be activated by the Near Field Communication (NFC) smart tag antenna in your cell phone to provide enough energy to switch a ferroelectric capacitor.

For this project, we will use the LTD0-028K

## RESOURCES

Tester website  
[www.ferrodevices.com](http://www.ferrodevices.com)

Experimentalist website  
[www.ferromems.com](http://www.ferromems.com)

Vendor website for the microprocessors  
<http://arduino.cc>

Measurement Specialties, Inc.  
<http://meas-spec.com>

## Event Detection

The Type AD ferroelectric capacitor used in the February article to demonstrate the memory circuit of **Figure 2** required six nanocoulombs of charge to be delivered within 4.2 volts. (Refer to the vertical axis of

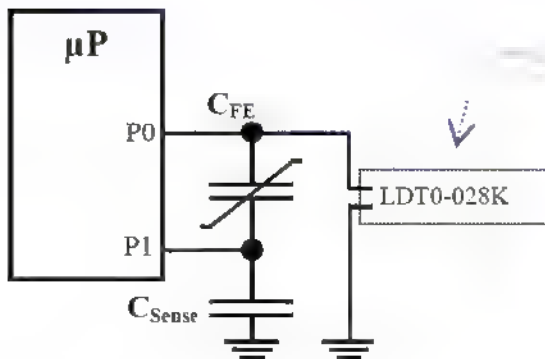


FIGURE 3. Simple event detector circuit and the LDT0-028K sensor.



FIGURE 5. An event detector circuit on a perboard.

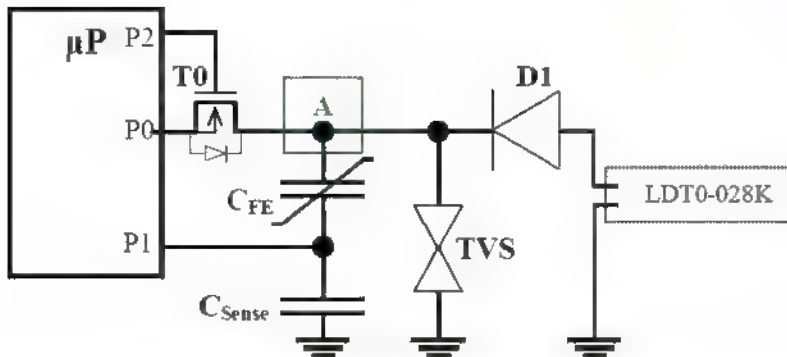
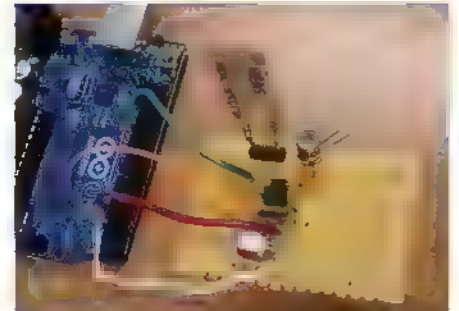


FIGURE 4. An event detector circuit complete with overvoltage protection.

piezoelectric flapper manufactured by Measurement Specialties, Inc. The LDT0-028K is labeled as a vibration sensor or as a piezoelectric switch. It consists of Polyvinylidene Fluoride (PVDF) — a piezoelectric plastic with metallic electrodes. If its plastic tab is bent over 90 degrees, the LDT0-028K will generate up to 7 nC of charge, or 90 volts. In **Figure 3**, we use a plastic piezoelectric capacitor to switch a ceramic ferroelectric capacitor in an event detector circuit.

There are issues using the LDT-028K bender as shown in **Figure 3**. It is a capacitor, so when sharing charge with the Type AD ferroelectric capacitor, it will generate the voltage set by the ferroelectric capacitor for the amount of charge being deposited. If the ferroelectric capacitor starts DOWN, a bend of the plastic piezoelectric switch generates 7 nC to force the capacitor UP. According to **Figure 2**, the voltage will only reach about six volts.

However, if the ferroelectric capacitor is already UP and does not switch, the switch could generate a far higher voltage across the ferroelectric capacitor and damage it. Even more important, if the plastic piezoelectric switch is bent the wrong way, it will generate negative voltages!

Another issue with the simple circuit in **Figure 3** is the status of P0 and P1 when the microprocessor is powered down. All pins on a microprocessor package will have clamping diodes to the power rails (Vcc and ground) to dissipate static discharge inadvertently applied to a pin.

When the LDT0-028K attempts to generate a voltage on P0 to switch the ferroelectric capacitor UP while the microprocessor is powered off, the clamping diodes on P0 will limit the voltage range to  $\pm 0.7$  volts.

A blocking transistor must be placed on P0 to isolate the external components from the clamping diodes of P0 inside the package when the microprocessor is turned off. On the other hand, we can use this property to our advantage on P1. When the piezoelectric switch attempts to change the state of the ferroelectric capacitor and the microprocessor is off, the ferroelectric charge will flow into C<sub>Sense</sub> until the

voltage on C<sub>Sense</sub> exceeds the clamping voltage on P1; whereupon, the excess current will flow to ground. We will not have to worry about back voltage from C<sub>Sense</sub> fighting the voltage across the ferroelectric capacitor applied by the switch. To solve these issues, we add three components to the event detector circuit in **Figure 3** to create a realistic real world circuit in **Figure 4**.

T0 isolates the common node A from P0 and is only enabled by the microprocessor when it wants to write or read C<sub>FE</sub>. When power is off to the microprocessor, T0 will be off to allow the piezoelectric switch (LDT0-028K) to generate a voltage across the ferroelectric capacitor and not be clamped to ground by P0. When power is on, event detection will still function.

T0 is turned off during detection. P0 and P1 are left at zero volts. Sensing pulses from the LDT0-028K are blocked from P0 by T0, but current from C<sub>FE</sub> can flow into P1. D1 — at all times — blocks negative voltages from the sensor, and the TVS limits the voltage across the ferroelectric capacitor to no more than +6 volts.

Recommended components for the detector are:

1. Radiant Technologies Type AD 10,000  $\mu\text{m}^2$  ferroelectric capacitor
2. 680 pF C<sub>Sense</sub>
3. 2N7002 N channel FET for T0
4. SMAJ-6.0A for the TVS
5. 1N914 for D1

Our prototype of the circuit appears in **Figure 5**. The piezoelectric flapper is mounted directly on the board in **Figure 5** for debug purposes. Later, it can be mounted to a door so that its tab will be bent by the door when the door is opened. The sensor is connected to the circuit board using twisted pair.

Note that a voltage limit to  $C_{FE}$  is introduced by T0. P2 will activate the gate of T0 to turn it on. As  $C_{FE}$  charges up during a *read* operation, it will stop 0.7 volts below the gate voltage. This will not be a problem in our recommended circuit as the Arduino Uno I/O pins operate at five volts and the Type AD capacitor saturates at 2.1 volts. It does become a problem for thicker ferroelectric capacitors like the Type AB. The voltage drop across the transistor disappears when  $C_{FE}$  is being written down by P1, while P0 is ground.

## Programming

The program to operate the event detector from the Arduino is similar to that explained in the February article, but more automatic. Only one routine is needed. The routine always executes a *read* followed by a *write DOWN* to reset the event detector for the next detection period:

1. Set P2 high to turn on T0.
2. Leave P1 as an input to detect the voltage across  $C_{Sense}$ .
3. Set P0 high.
4. Read the digital state of P1.
  - a. If  $C_F = \text{DOWN}$ , P1 will show Logic 1 indicating that no event occurred.
  - b. If  $C_F = \text{UP}$ , P1 will show Logic 0 indicating that the event occurred.
5. Set P0 low.  $C_{FE}$  and  $C_{Sense}$  will both be at zero volts.
6. Set P1 high to write  $C_{FE}$  DOWN to prepare for the next detection period.
7. Set P1 low.  $C_{FE}$  and  $C_{Sense}$  will again both be at zero volts.
8. Set P2 low to turn off T0 and isolate P1.

The next power-on detection period now starts. If power is to be turned off, first set both P0 and P1 as inputs to ensure no voltage spikes are applied to  $C_{FE}$  during power-down when the logic inside the Arduino goes unstable as voltage drops below the minimum specification.

The Arduino can be set up to execute the *read-reset* routine from one of three situations:

1. Subroutine call from a master program.
2. Upon reset by the reset button on the Uno.
3. Upon power-up.

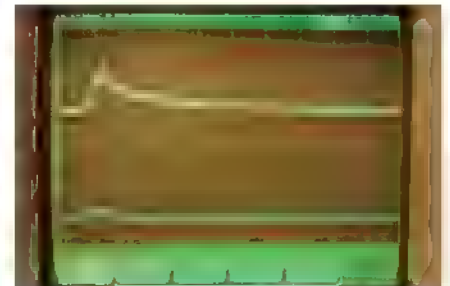
The program to operate the Uno and the event detector is located at the article link. The program turns on the LED on the Arduino board attached to pin 13 to indicate if an event occurred.

## The Event Detector in Operation

It is important to remember when mounting the LTD0-028K flapper on the door that it can generate either a positive or negative voltage. The direction can be determined with an oscilloscope connected to the event detector circuit at point A.

An activation of the LTD0-028K flapper causes the voltage spike across  $C_{FE}$  shown as the top trace in **Figure 6**.

The vertical scale is two volts per grid line. **Figure 7** captures the voltages at A



**FIGURE 6.** Voltage generated across  $C_{FE}$  (top trace) by the piezoelectric flapper bent 90 degrees.

## Feedback Motion Control

### The Old Way

- 1) Build robot
- 2) Guess PID coefficients
- 3) Test
  - 3a) Express disappointment
  - 3b) Search Internet, modify PID values
  - 3c) Read book, modify PID coefficients again
  - 3d) Decide performance is good enough
  - 3e) Realize it isn't
  - 3f) See if anyone just sells a giant servo
  - 3g) Express disappointment
  - 3h) Re-guess PID coefficients
  - 3i) Switch processor
  - 3j) Dust off old Differential Equations book
  - 3k) Remember why the book was so dusty
  - 3l) Calculate new, wildly different PID coefficients
  - 3m) Invent new, wildly different swear words
  - 3n) Research fuzzy logic
  - 3o) Now it is certainly not working in uncertain ways
  - 3p) Pull hair
  - 3q) Switch controller
  - 3r) Re-guess PID coefficients
  - 3s) Switch programming language
  - 3t) Start a new project that doesn't need feedback control
  - 3u) See parts in box. Feel guilty. Go back to old project
  - 3v) Start testing every possible combination of PID coefficients
  - 3w) Apply eye drops to red, binary, sleep-deprived eyes
  - 3x) Walk it's working!
  - 3y) Decide not to do any more projects that require control systems
  - 3z) Wonder why someone doesn't just make a thing that tunes itself

### The Kangaroo x2 Way.

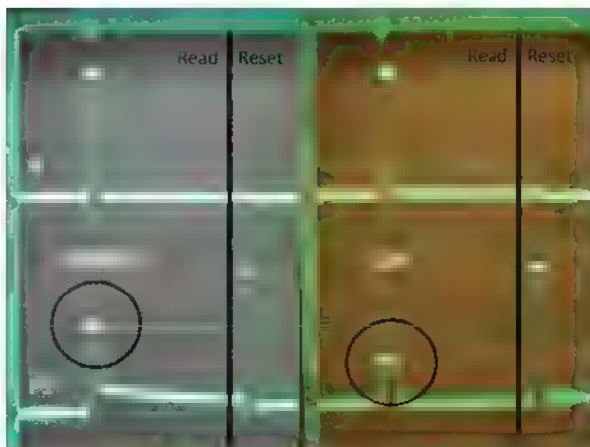
- 1) Build robot
- 2) Press Autotune
- 3) Get a snack

**Kangaroo x2 adds self-tuning feedback to SyRen and Sabertooth motor drivers.**

**\$24.99**

**DimensionEngineering**

[www.dimensionengineering.com/kangaroo](http://www.dimensionengineering.com/kangaroo)



**FIGURE 7.** Oscilloscope traces for event and no-event read-reset operations on the event detector.

and P1 during the *read-reset* operation for *no-event* (left) and *event* (right). To capture the traces, the *read-reset* routine was executed twice. The first execution preset  $C_{FE}$  DOWN. The outcome of its *read* operation was ignored.

The second execution determined if we had activated the piezoelectric flapper or not since the first execution.

The bottom trace in **Figure 7** shows the sense voltage on P1. If no event occurs,  $C_{FE}$  is still DOWN when the capacitor state is read. The ferroelectric switching charge will generate over three volts on the P1 pin from the charge in  $C_{Sense}$ . This voltage will be read as Logic 1 by the Arduino.

If the event occurred during the detection period, no ferroelectric switching will take place during the *read* operation; less charge will flow from  $C_{FE}$  to  $C_{Sense}$ ; less than two volts will be generated on the input P1. This state will read as Logic 0 by the Arduino.

## Conclusion

The ferroelectric event detector adds excitement to a single bit of non-volatile memory: An outside sensor performs the *write* operation so the microprocessor can detect the occurrence of events in its local environment. The event detector will remember the event after it occurs so the controller can be powered off while it waits.

A future article will eliminate the microcontroller altogether to make a stand-alone memory circuit out of discrete components. **NV**

# HACKABLE BADGES

OPEN SOURCE HARDWARE

Display your name (or your logo) and play interactive games with other badges.  
[WWW.PARALLAX.COM/BADGE](http://WWW.PARALLAX.COM/BADGE)

**PARALLAX**  
[www.parallax.com](http://www.parallax.com)

Learn at Home!

## Electronics Courses

Cleveland Institute of Electronics

Train at home to be a professional *electronics* or *computer* technician. Learn with hands-on labs, instructor support, online exams, videos and instructor led chat rooms.

**FREE** course catalog [www.cie-wc.edu](http://www.cie-wc.edu) or (800) 243-6446

- NEW! Robotics Automation Lab
- NEW! Computer Security Specialist
- Industrial Electronics with PLC
- Broadcast Engineering
- Electronics with FCC
- PC Troubleshooting
- Electronics Troubleshooting
- Networking

Visit [www.cie-wc.edu](http://www.cie-wc.edu) and start today!

CIE: 1776 E. 17th St., Cleveland, OH 44114 • (800) 243-6446 • Registration 70-1-0002H

# Vintage Computing

## I Still Adore My 64

By Jim Lawless

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?magazine/article/october2015\\_Lawless](http://www.nutsvolts.com/index.php?magazine/article/october2015_Lawless)

*Although it has been several years since I got rid of my eight-bit Commodore equipment, my fondness for the Commodore 64 home computer has never subsided. In 1994, I discovered that I could run a program on my Windows PC that would emulate a C64. I have been happily tinkering with emulator software ever since.*

After using an emulator for only a short while, I found that developing software for the C64 could be a slow-going process. I managed to find a C-128 system that allowed me to buy compilers still available on 1541 disk media. I used the system to transfer code back and forth between my PC and the C-128. The higher-level Pascal and C compilers for those eight-bit systems weren't the easiest to use, often requiring one to exit an editor so that the compiler could load into memory and work with the source file from the disk.

I later discovered CC65 – a small C cross-compiler for 6502-based systems. CC65 is capable of producing a finished executable file for the C64 (as well as numerous other eight-bit computers and gaming systems). CC65 runs as its own processor or under most modern operating systems. This allows one to keep an editor, compiler console, and emulator session open in their respective windows to allow rapid edit/compile/execute operations.

### Resources

The VICE Project Repository  
<http://vice-emu.sourceforge.net>

The CC65 Website  
<http://cc65.github.io/cc65>

The Lemon64 Forums  
[www.lemon64.com/forum/index.php](http://www.lemon64.com/forum/index.php)

### Installation

The tool we'll use to emulate the C64 is known as VICE (Versatile Commodore Emulator). We'll use the Windows-specific port known as WinVICE. We'll also use a Windows port of the CC65 cross-compiler. If you use an operating system other than Windows, you will need to install VICE and CC65 for your given environment.

For Windows systems, we have packaged the WinVICE and CC65 together at the article link so that you can more quickly begin writing and compiling code.

If you extract the cc65vice.zip archive file to the root folder of your C: drive, you should see the following directory hierarchy:

```
c:\c64\cc65
c:\c64\dev
c:\c64\drive8
c:\c64\winvice
```

To invoke the WinVICE emulator for the C64, double-click on X64.EXE in the c:\c64\winvice folder. You should see the introductory C64 screen, followed by a BASIC "READY" prompt. Let's type in a short BASIC program. Note that the keyboard follows C64 key positioning conventions. The equals sign "=" will appear when you press the backslash key "\":

```
10 FOR X 1 TO 10
20 PRINT X
30 NEXT X
RUN
```



FIGURE 1. A BASIC for loop.



Refer to **Figure 1**. The emulated drive 8 (as I have the system configured) is the folder `c:\c64\drive8`. If you look in this folder via Windows, you'll see a handful of files there. To see the same list from within WinVICE, type the following:

```
LOAD "$",8
```

Please note that the double-quotation mark character is entered by typing SHIFT-2 on the emulator keyboard.

The directory listings from Commodore eight-bit machines load into memory like BASIC programs by using the special filename "\$." You can then type LIST to see the formatted listing as in **Figure 2**. Please note that this folder can be changed by clicking on the "Settings" menu. Then, click the "Peripheral Settings" submenu item to change the location.

## Going Old School

VICE has its own machine-language monitor that you can invoke externally from the running C64 environment.

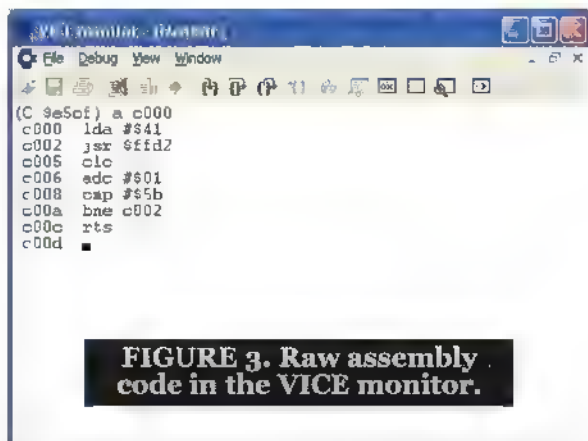


FIGURE 3. Raw assembly code in the VICE monitor.

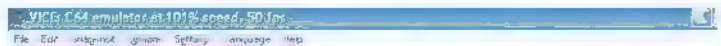


FIGURE 2. A directory listing.



Click "File" and then click "Monitor" from the top menu. You can enter a short machine-code subroutine that will print the alphabet by iterating over the PETSCII alphabet characters, calling the *CHROUT* character-output routine at `$FFD2` on each iteration.

Enter these lines at the monitor prompt:

```
a c000
lda #$41
jsr $ffd2
clc
adc #$01
cmp #$5b
bne $c002
rts
```

Refer to **Figure 3**. Next, close the monitor window. From the C64 BASIC "READY" prompt, enter the line:

```
SYS 49152
```

The above line will invoke the subroutine we just wrote at location 49152 (`$C000` hex). Refer to **Figure 4**.

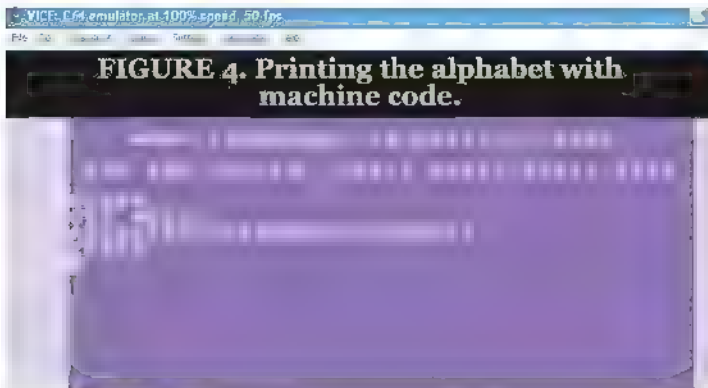
## C Programming — The CC65 Cross-Compiler

CC65 began as a port of the Small C Compiler for the Atari eight-bit family. Many extensions were added in addition to support for other systems.

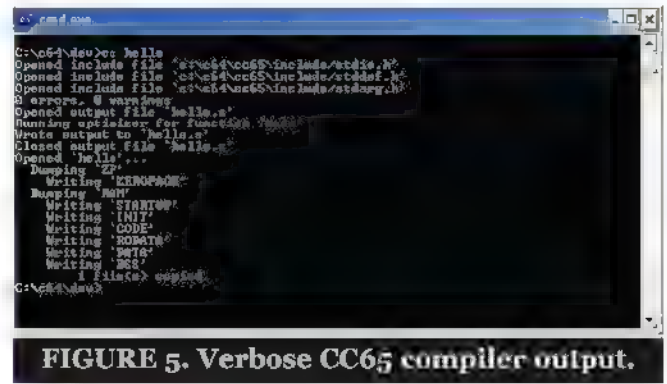
Writing C in CC65 can be a little different than writing C for other systems. It's more optimal to deal with unsigned eight-bit data when possible. While I try to do this in my code using the "unsigned char" data type for counters, I often just use the *int* data type, which is a bit less efficient.

The goal of the programs presented here isn't to show CC65 optimization skills.

Let's use the traditional "Hello, world!" program as



**FIGURE 4. Printing the alphabet with machine code.**



**FIGURE 5. Verbose CC65 compiler output.**

our first. Here is the source code for the program (hello.c):

```
#include <stdio.h>
int main()
{
    printf("Hello, world!\n");
}
```

From a command prompt, change to the C:\c64\dev directory. We have provided a batch file called "CC.BAT" that will be used to compile and link the given C program. It will then copy the executable file to C:\c64\drive8 so that it can be loaded by the emulator.

From the command prompt, enter the following:

```
cc hello
```

You should see something like the screen depicted in **Figure 5**. From a WinVICE window, load the program from drive 8 and run it. Refer to **Figure 6**.

Note that the startup code that CC65 embeds into every executable program forces the C64 display into "lower case" mode. CC65 has also automatically translated our ASCII constant "Hello, world!" into PETSCII so that we don't need to convert back and forth ourselves.

If you'd like to look at or change any of the code in hello.c, issue the following from a command prompt:

```
notepad hello.c
```

You can then save hello.c after making your changes. You can invoke the "cc" batch file again to compile your changes, and copy the executable program to WinVICE's drive8 directory.

Let's try a version of hello.c that takes advantage of the C64 colors. Here's colorhello.c:

```
#include <stdio.h>
#include <peekpoke.h>
int main()
{
    // set border to black (53280)
    POKE(0xd020, 0);

    // set background to black (53281)
```

```
POKE(0xd021, 0);

    // set text color to white
    POKE(646, 15);

    printf("Hello, world!\n");
}
```

This program adds an *include* directive for the header file "peekpoke.h." This header file contains macros that make it easier to deal with fetching and storing eight-bit and 16-bit values from/to memory. The POKE macro we used above looks like this:

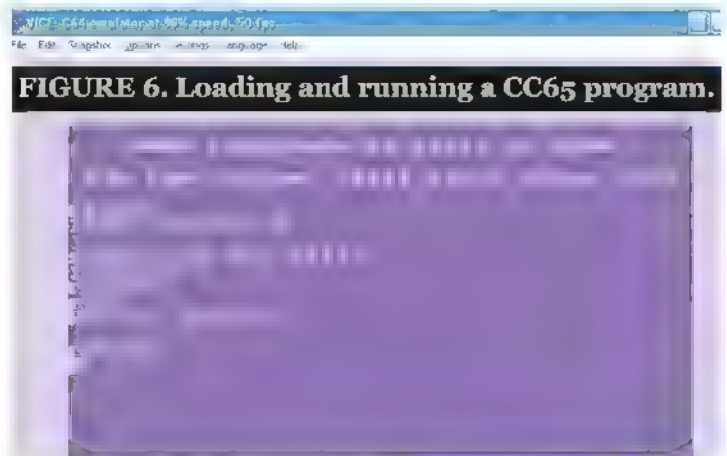
```
#define POKE(addr, val) (*(unsigned char*)
                        (addr) (val));
```

We could have written the first expression as:

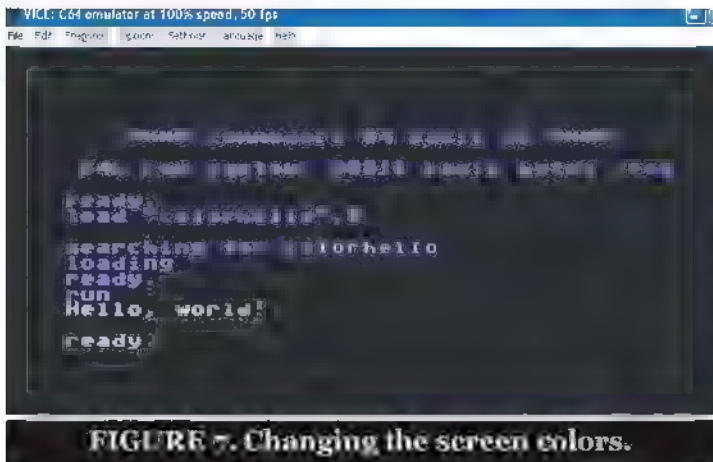
```
*(unsigned char *) (0xd020) 0;
```

The POKE macro provides syntactic sugar for an expression like the above.

You can take a look at the peekpoke.h header file or any of the header files in the C:\c64\cc65\include folder. You'll need to use an editor that can understand Unix-style bare linefeed characters to terminate a line.



**FIGURE 6. Loading and running a CC65 program.**



The `colorhello.c` program first sets the border and background colors to black. I used the hexadecimal constants `0xd020` and `0xd021` because any number over 32767 expressed in decimal is assumed to be a “long” by the compiler.

Note that I used a decimal constant for location 646 which contains the current character color.

When you load and run the `colorhello` program, you should see something like the screen in **Figure 7**.

If you try to LIST the program, you’ll see a line similar to the following:

```
704 sys2061
```

The above is a BASIC stub that’s just long enough to invoke the compiled machine-language payload that follows it in memory.

## Fun and Games

Part of the fun of owning a microcomputer in the 1980s was typing in games from listings in books and magazines. Let’s take a look at a few simple game-related programs.

Most single-player games require the use of a random number generator function. CC65 provides the `rand()` function which returns a random unsigned integer in the range of 0 to 32767 inclusive. The randomizer should be seeded by the `srand()` function at the beginning of a program to ensure that the series of random numbers will be different each time the program is executed.

Let’s take a look at `random.c`. From a command prompt in the `C:\c64\dev` directory, enter the following:

```
notepad random.c
```

The `random.c` code isn’t a game by itself, but it illustrates a simulation of dice rolls. In addition, it illustrates a few CC65 concepts that we can study.

One of the earliest things we do in the code is to see the random number generator with this line of code:

```
srand( PEEKW(0x00a1));
```

The “jiffy” clock that is updated by interrupt routines is stored at locations `$A0`, `$A1`, and `$A2` in zero page memory. This is the clock used by the BASIC meta-variable `TIME$`. In the line of code above, I use the last two bytes of the clock as an unsigned `int` initializer for `srand()` since we’re unlikely to have the same values there each time we run the program.

`Random.c` contains two function prototypes near the beginning of the program:

```
int roll_one_die(int);
int roll_dice();
```

This allows functions that appear earlier in the code to understand the parameter list requirements and return values of each function.

Our function `roll_one_die()` accepts an integer parameter. This parameter is named “sides” in the body of the function code itself. The variable `side`’s visibility is limited to use within the function `roll_one_die()`. Parameters are local variables.

The function `roll_one_die()` is intended to simulate a single die roll. Many role-playing games use dice with varying numbers of sides. In the code, we call `roll_one_die()` with a parameter of 6 when we want to simulate rolling a six-sided die. We call the same function with a parameter of 20 when we want to simulate a 20-sided die.

The function `roll_dice()` simulates rolling two six-sided dice. Note that it calls `roll_one_die()` twice and then returns the sum of those values.

Also note that in the body of `main()`, I issue a “return” command with the value zero. I did not do this in earlier examples and received no warnings from the compiler. It’s good form to supply the return value from `main()`.

Let’s compile and run `random.c`. From a command prompt, enter the following:

```
cc random
```

You should see the verbose compilation messages.

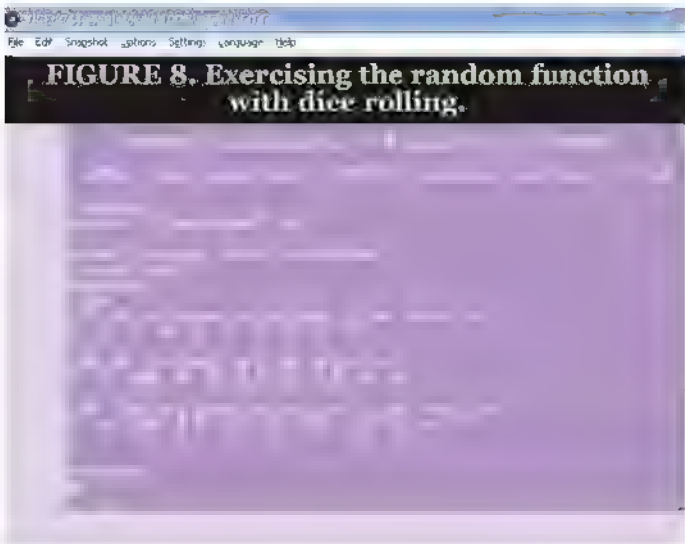
Then, load and run “random” from WinVICE. You might see output similar to that in **Figure 8**.

If you run it again just by typing the word “run” and enter, you should see different sets of random dice rolls. Let’s take a look at `guess.c`:

```
notepad guess.c
```

This program uses a function named `play_the_game()` as the primary function. It is called from `main()` in a





**FIGURE 8. Exercising the random function with dice rolling.**

conditional loop that executes (at most) once. The function *play\_that\_game()* uses a few local variables. The variables “number,” “tries,” and “answer” are all integer variables. The variable “answerBuff” is an array of characters. I left enough room in the buffer to hold up to three digits of entered text, a linefeed, and a null byte as a string terminator.

The *play\_that\_game()* function uses a “forever” for loop (one with no iteration condition specified). The code explicitly uses the “break” verb to drop out of the loop once the user guesses the correct number.

Compile *guess.c* and execute it:

```
cc guess
```

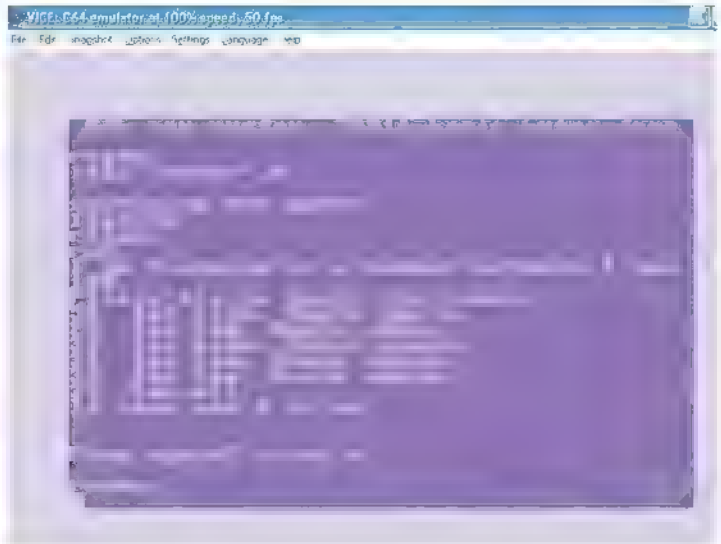
Refer to **Figure 9**. Since the values chosen are random, your game session will probably look a bit different than the one here.

## Pointers, Data Structures, and Dynamic Memory

CC65 allows for use of C data structures via the “struct” keyword. Let’s consider using a struct to describe a room in an adventure game. For our needs, a room will simply have a description and will then have up to four references (pointers) to other room structs. Each of the pointers will refer to a direction to the North, South, East, or West of the current room.

The struct definition in our program *rooms.c* will look like this:

```
struct room {
    struct room *north;
    struct room *south;
    struct room *east;
    struct room *west;
    char *description;
} *r1,*r2,*r3, *r4,*the_exit;
```



Note that we have created five global variables (*r1*, *r2*, *r3*, *r4*, and *the\_exit*). All of them are pointers to a data structure named “room.”

In the *init\_rooms()* function, we will allocate memory for each of the above by invoking the *malloc()* function:

```
the_exit (struct room *) malloc(sizeof(struct
room));
the_exit->description
    "the outside of the building";

r1 (struct room *) malloc(sizeof(struct
room));
r1->description
    "the main lobby";

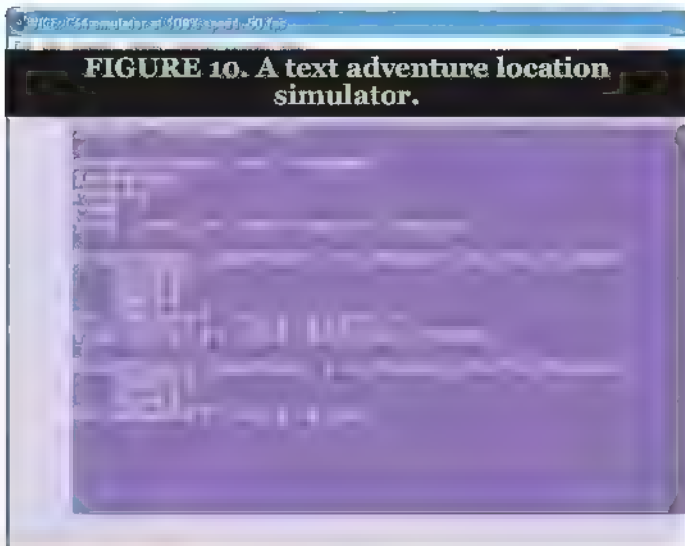
r2 (struct room *) malloc(sizeof(struct
room));
r2->description
    "the dining room";
```

Later in the code after all are allocated, we tie the directional pointers to the appropriate rooms:

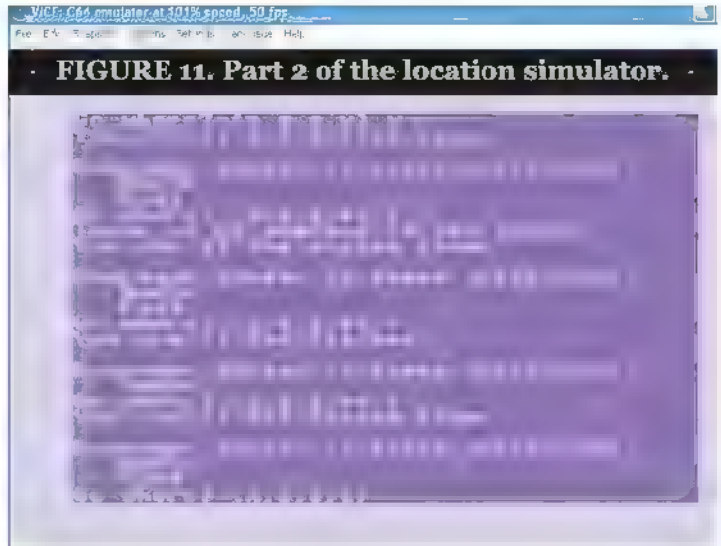
```
r1->south the_exit;
r1->north NULL;
r1->west r2;
r1->east r4;
```

Our *look()* function displays the current room data structure’s “description” string. The function then looks at each of the four directional pointers. If any are not NULL, that direction will show up as a visible “doorway” to another room.

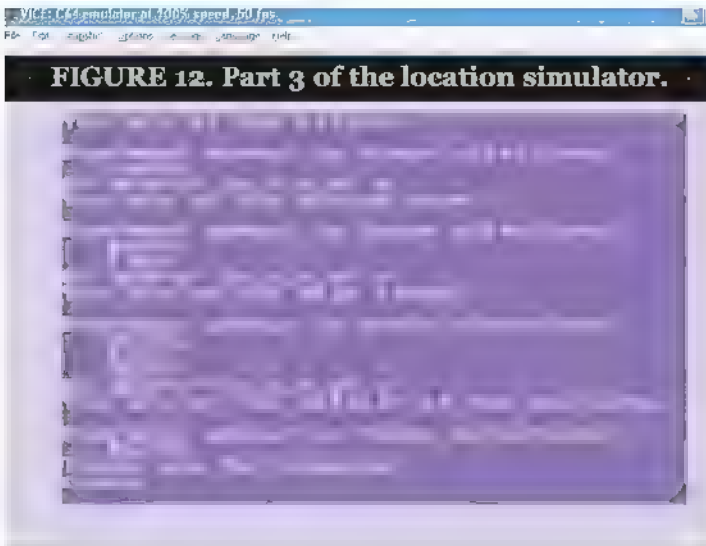
Jim Lawless has been programming professionally for nearly 28 years. He wrote for mainstream magazines such as *Dr. Dobbs Journal* and *The C++ Users Journal*. He has been an adjunct faculty member of a local community college, specializing in teaching computer programming languages. His current specialty skills relate to JavaScript, Java, and .NET technologies.



**FIGURE 10. A text adventure location simulator.**



**FIGURE 11. Part 2 of the location simulator.**



**FIGURE 12. Part 3 of the location simulator.**

Our *main()* function calls the initialization function and then iterates in an *input* loop. Once you have headed South from the “main lobby,” you have walked outside of the building, ending the simulation.

Here’s a quick walkthrough of the rooms program. (Refer to **Figures 10, 11, and 12.**) The data structures could be fleshed out with other data structure references. Perhaps each room should contain a pointer to a list of items to be found there. We would want to create data structures to model the player, as well as the rooms. A player data structure might have strength, stamina, and intelligence attributes, as well as a list of items they are carrying and a reference to the room they are in.

Note that we created the data structures themselves from allocated memory blocks that live on the “heap.” Although we don’t have a whole lot of memory to work with, the heap is a managed structure that can allow dynamic allocation and the freeing of memory.

To free up a memory block allocated with *malloc()* or

one of the related functions, one can call *free()* passing the pointer to the block of memory as the argument.

If you need a very large block of memory as a buffer for a file copy operation or something similar, the only way I’ve found to allocate memory in a large chunk is to try for something absurdly large and decrease the requested amount until you succeed. Refer to **Figure 13** to see the output of the *mem.c* program.

I would recommend leaving some room so that the big allocation doesn’t limit our stack usage for automatic local variables and return addresses.

A C64 program written with CC65 uses memory in the range \$0800 - \$CFFF inclusive. The BASIC ROM at \$A000 is disabled so that the 8K of RAM underneath can be utilized. The C runtime stack begins at location \$CFFF and works backward in memory. The heap begins at the end of the program code and grows toward the stack space.

In addition to the *char* and *int* types we’ve used so far, CC65 natively supports signed and unsigned long integers. The program *longs.c* demonstrates the usage of a long variable and a long constant.

## Commodore-Specific Functions and Structures

The header file *cbm.h* and the corresponding library provide Commodore-specific functions and data structures to the CC65 compiler. Please consider the example in *showdir.c*.

In *showdir.c*, we open a directory using the CBM-specific *cbm\_opendir()* function. We pass in the address of a *cbm\_dirent* data structure. If the function returns zero, we were successful and the data structure is filled.

The *while-loop* then displays each directory entry name, the number of 256 byte blocks consumed on the disk, and the CBM file type. Refer to **Figure 14**.



**FIGURE 13. Allocating a large block of memory.**



**FIGURE 14. Displaying a directory with CC65.**

Note that this looks very similar to the directory listing we saw via `LOAD "$",8` at the beginning of this text.

and CC65. If you'd like to discuss either further, please feel free to contact me at jimbo@radiks.net.

Please remember to un-spam that address in your email program so that I can reply. **NV**

## SYS 64738

I'm afraid that's the end of the introduction to VICE

**SUPERIOR EMBEDDED SOLUTIONS**



**DESIGN YOUR SOLUTION**  
CALL (480) 837-5200

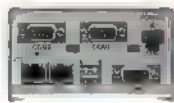
[www.embeddedARM.com](http://www.embeddedARM.com)



### TS-7250-V2 Single Board Computer

Extensible PC/104 Embedded System with Customizable Features and Industrial I/Os

- 800 MHz or 1 GHz Marvell PXA166 ARM CPU
- 512 MB DDR2 RAM
- 2 GB SLC eMMC Flash Storage
- PC/104 Connector
- 8k or 17k LUT FPGA
- Dual Ethernet



Available with TS-ENC720 enclosure

**Starting at**  
**\$165**  
Qty 100  
**\$199**  
Qty 1



### PC/104 Peripherals

#### Multi-Function Daughter Cards

Our Line of PC/104 Peripheral Boards Offer Maximum Functionality at Minimum Cost.

- Digital I/O Boards
- Ultra Long Range Wireless
- Software Controlled Relays
- Multi-tech GSM, GPRS, HSPA modem boards
- Intelligent battery back-up
- Much more....

**Starting at**  
**\$32**  
Qty 100  
**\$30**  
Qty 1



Longwave discontinued a product in 30 years



Embedded systems that are built to endure



Support of the open



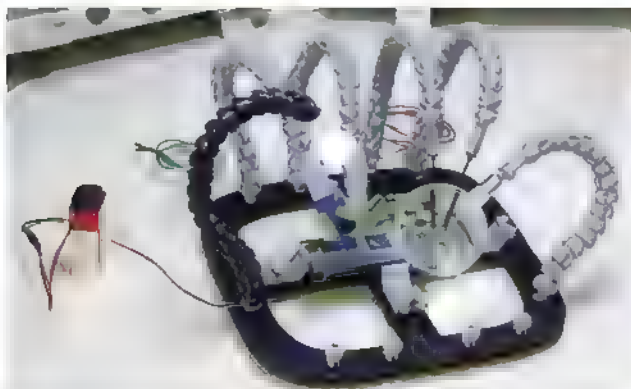
Unique embedded solutions add value for our customers

*i 3 p nter o  
pr tic l proje s on  
yo r ork e c*

# 3D Print Designs for Electronic Hobbyists

I've shown in previous articles that with a home 3D printer, you can print many kinds of tools for your electronics workshop. One place I like to look for ideas is on Thingiverse.com to see what others are producing. Once in a while, I come across 3D prints that are nothing short of creative genius. Here are a few I recommend you check out.

## PCB Workstation with Articulated Arms



■ FIGURE 1. PCB Workstation with Articulated Arms.

One of the designs I found was a **PCB Workstation with Articulated Arms** submitted by Thingiverse user **giufini**. Most of the designs on Thingiverse are open source so you are free to print and use them as you wish.

There are so many useful 3D prints in this one design. The adjustable arms are really clever and seem to be potentially useful for multiple different ways to hold a probe or lead from test equipment.

This particular 3D print has many pieces to it so it will take some time to print everything, but you can start by building a basic unit and then add articulating arms at any time. This is only limited by your imagination.

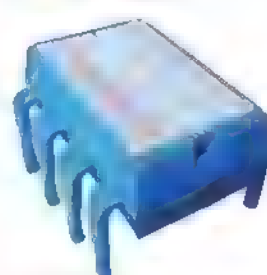
It takes the popular "third hand tool" that is often used to hold objects while you solder them and gives you a fourth, fifth, and sixth hand, and possibly more when you are trying to debug your electronics.

## Breadboard Project Box



■ FIGURE 2. Breadboard Project Box.

Another design that caught my eye was the **Breadboard Project Box** by Thingiverse user **wgbartley**. This is similar to the large Integrated Circuit Box I created earlier in this column series, but gets rid of the IC legs and



■ FIGURE 3. My Integrated Circuit Box. still have some storage for parts.

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php??magazine/article/october2015\\_Practical3DPrinting](http://www.nutsvolts.com/index.php??magazine/article/october2015_Practical3DPrinting).

## Filaments



■ FIGURE 4. Conductive Filament.

One of the more exciting areas for electronics and 3D printing is the new filaments available from various sources. At the top of the list is **conductive filament**. This is plastic infused with conductive material. Now, this isn't as

conductive as a copper trace of a circuit board, but is similar to the conductive properties of conductive paint or a conductive pen. This type of filament is great for making plastic boxes that help protect the electronics inside from electromagnetic interference if the box is connected to a ground at multiple points. It's certainly not as good as a metal box, but does offer that advantage of custom plastic shapes with some resistance to outside interference.

The conductivity of these plastics varies from manufacturer to manufacturer. Typically, the resistance spec is based on a measurement of conductivity through a 1 cm x 1 cm x 1 cm block. This is often reflected as an ohms-cm or ohms/cm value. I've seen everything from 1,200 to 10,000 listed in the various filament specifications, so you may have to be selective when choosing a conductive filament.

I bought the low cost conductive filament shown in **Figure 4** at my local Microcenter. It was \$22 for a 1 kilogram role. I've printed a few conductive elements that I could actually use to control the touch screen of my iPhone. I hope to produce some boxes for electronics and see how it compares to a metal box.

## 2020 Beams



■ FIGURE 5. 2020 Plastic Beams.

Another design I've started playing with are 2020 extruder beams. You can get these in aluminum just about anywhere now, but I wanted to see if I could print them in plastic as this would give me more options for lighter weight designs and also allow me to make custom versions that may not be possible with aluminum.

They won't be nearly as strong as the aluminum versions, but I'm thinking I can use them for mounting things on my bench or building simple test fixtures. If I use conductive plastic, I might be able to isolate wires that run through the center hole from interference. The possibilities are endless.

I found a design on Thingiverse for a 2020 size beam from user **beverageexpert**. I downloaded the .STL file and printed a few samples. I created some 90 degree brackets in Tinkercad but the tolerances were a bit tight. I ended up breaking some of the beams just trying to connect them.

All these were printed with ABS plastic and they are stronger than I expected, but they tend to split at the layer lines if too much force is applied. I can use a little acetone or Super Glue™ to fix them anytime they break or to make them into a permanent structure. Acetone works great to fuse ABS pieces together.

The only limitation to all this is the size of my 3D printer. I can't print larger than seven inch beams, so I'll have to create some connectors so I can connect beams to create larger than seven inch sizes. In fact, I printed only six inch beams to start

## Conclusion

Your 3D printer can become a great addition to your set of electronics tools if you let your imagination and the imaginations of many others loose. Through the wonderful world of open source 3D printed designs, most of the work is already done for you. You just need to 3D print it. **NV**

## Resources

Check out my website and blog:

[www.elproducts.com](http://www.elproducts.com)

YouTube Channel:

[www.youtube.com/user/beginnerelectronics](http://www.youtube.com/user/beginnerelectronics)

My 3D designs:

[www.thingiverse.com/elproducts/designs](http://www.thingiverse.com/elproducts/designs)

Tinkercad:

[www.tinkercad.com](http://www.tinkercad.com)

**2020 Beam:** [www.thingiverse.com/thing:280318](http://www.thingiverse.com/thing:280318)

**Breadboard Box:** [www.thingiverse.com/thing:930427](http://www.thingiverse.com/thing:930427)

**PCB Workstation:** [www.thingiverse.com/thing:801279](http://www.thingiverse.com/thing:801279)

# GPSL 2015 and my 150th Near Space Launch

**The Great Plains Super Launch (GPSL) is a conference of amateur near space explorers that has been meeting since 2001. In that time, I've launched 125 near space missions and carried over 300 BalloonSats for students. With a little help from my friends, I was able to send four more balloons skyward at GPSL 2015 and I would like to tell you more about them.**

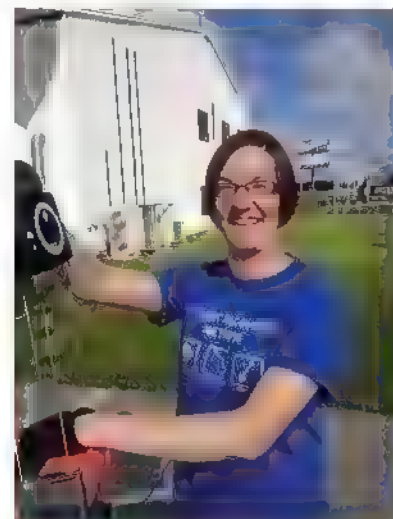
**W**ow, what a summer! As a high school teacher, I get my summers off. Often, I'm working a summer job or attending conferences, but I do get a chance to travel and partake in fun events. One of the events I always look forward to is the Great Plains Super Launch, or GPSL. This year, the Missouri Area Robotics Society (ROBOMO) hosted GPSL at the Missouri Baptist University in Moscow Mills.

For readers not familiar with it, GPSL is a conference for amateur near space explorers and attended primarily by amateur radio balloon groups located in the Midwest. The conference began in 2001 in an effort to fly three near space missions for *Weatherwise* magazine (see the article "Near-Space Race" on page 14 of the

November/December issue). Every summer, a different group hosts the conference and we get to see some of the local flavor. Going to Missouri for GPSL this year reminded Rachel and me of just how humid it gets in the Midwest during the summer — we're quite spoiled here in Idaho. GPSL 2015 ran from June 18-20 and some eight high altitude balloon groups gave and heard presentations, discussed the technology and procedures of near space flights, and met new and old friends.

## Friday: Quadcopters to the Rescue

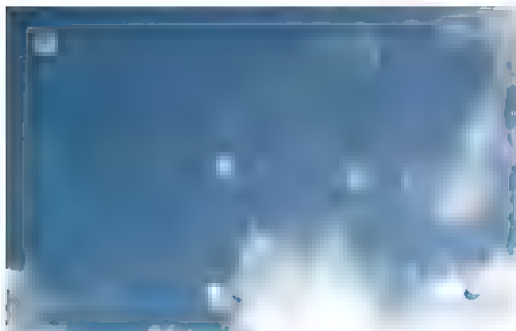
The presentation on quadcopters given by Atom of ROBOMO was a first for GPSL, and made me realize how they could be used in Near Space Search and Rescue (NS-SAR). One issue we occasionally deal with in amateur near space exploration is recovering a payload after it has landed in a region with no ham radio operators or I-Gate coverage. I-Gates — or Internet Gateways — are amateur radio stations that send the position reports of near spacecraft to the Internet where anyone — even those who are



**Figure 1.** Smiling Rachel, my wife helping with the launch of mission 148. Photograph by Jerry Gable (I believe).

not licensed hams — can see them. Since near spacecraft only transmit once per minute (usually) and descend at 1,000 feet per minute, missing position reports from the last few minutes of descent can increase the size of the recovery zone to search.

Sometimes, we luck out and can receive a position report over the radio as we enter the recovery zone. Other times, however, the near spacecraft will land in a gully or behind a hill that prevents chase crews from receiving a position report from a near spacecraft on the ground. Most near spacecraft report their positions using APRS (Automatic Position Reporting System) and transmit that information over VHF or UHF frequencies. As a consequence, unless there's a direct line of sight between the near spacecraft and a chase crew, no position reports can be received (that being said, there are some cases where signals can be refracted or reflected, and chase crews will get a position report from a hidden near spacecraft). In cases where chase crews don't receive position reports on the ground, they drive around the recovery zone hoping to get a visual contact. There's an equation I teach



**Figure 2.** Five of the six balloons launched at GPSL 2015.

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?/magazine/article/october2015\\_NearSpace](http://www.nutsvolts.com/index.php?/magazine/article/october2015_NearSpace).



**Figure 3.** Jeff Ducklow's near space module (onboard flight NearSys 15N) recorded this image of near space flight NearSys 15M at an altitude of 52,844 feet.

about calculating the distance to the horizon based on a titude that goes like this:

$$\text{Distance (miles)} = \sqrt{1.5 * \text{Height (feet)}}$$

Therefore, if you take your altitude in units of feet, multiply it by 1.5, and then take the square root, you'll calculate the distance to the horizon in units of miles. The beauty of this equation is that if chase crews can loft an amateur radio repeater to an altitude of 100 feet on a quadcopter, then it can get a line of sight to anything on the ground for 12 miles. This is a much greater range than chase crews can manage, and will make locating near spacecraft much less time-consuming. This is such an exciting concept that I plan to begin saving my pennies for a quadcopter and digital radio repeater

**Figure 4.** Another unusual picture. Most cameras don't have the wide angle lens of a GoPro to show nearly the entire near spacecraft like this during descent. Notice that the flight train is twisted; things get really chaotic during the early parts of descent. One reason for the chaos is the remains of the balloon hanging off the 15 foot load line attached to the apex of the parachute.



(digi-peater). If the concepts work as I think they will, I'll write a future article on this recovery method.

## Saturday: Launch, Chase, and Recover

Saturday morning was warm and humid, and only promised to get more sticky as the day progressed. Six teams were there to launch balloons, and included: ROBOMO; NearSys; Arch Reactor; Near Space Ventures; ARBONET; PENS; Jeff Ducklow; Bill Brown; Ann Boes (Columbia 4H); and Jerry Gable.

We were able to get five of the six balloons launched simultaneously and made sure to space the balloons far enough apart to avoid collisions during ascent (refer to **Figure 2**). Needless to say, two of the balloons did manage to collide and wrap around each other for less than a minute. This is the first time I've seen this happen in 152 near space missions.

With so many simultaneous balloon launches, it's not surprising for near spacecraft to record images of other balloons. Because of the differences in ascent speed, however, these images occur either at launch (when the balloons are still close to each other) or at high altitude where the balloons can be over a mile apart. You can imagine our surprise then when Jeff Ducklow's near space module recorded an image of a second near spacecraft that remained close all the way to an altitude of 10 miles. **Figure 3** is the first near space picture like this, as far as I know.

GoPro cameras have significantly

changed the kind of images returned from near space. One reason is their high resolution wide angle lens.

**Figure 4** was taken by Mike Moody's (of ROBOMO) GoPro and shows the near spacecraft early in its descent. You can see three modules below the red parachute. Attached to the top of the parachute is a load line connecting to the balloon. Most of the balloon didn't shear off during the burst and can be seen to the upper right of the image.

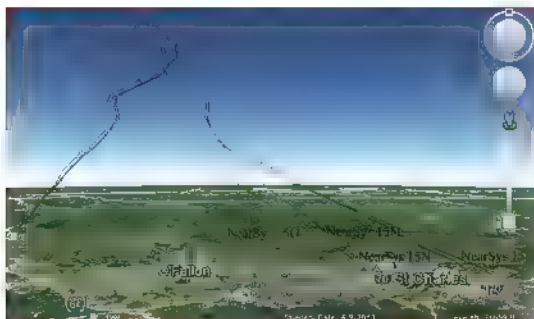
I want to thank ROBOMO for their hospitality during GPSL 2015 and a very special gift. As I mentioned, I launched my 150th near spacecraft at GPSL and in honor of the occasion, they created a commemorative piece of art. **Figure 5** shows the stainless steel disk cut with a water jet. (When I hang it in my house, I'll make sure to attach it to a stud in the wall and not just into drywall!)

## Sunday: Data Analysis

I collect some sort of science data on each of my missions, and I'd

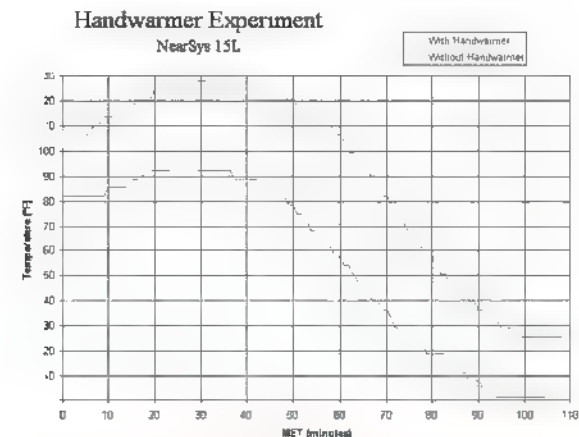


**Figure 5.** This three foot diameter commemorative piece was signed by everyone in attendance at GPSL 2015. It was great flying with you guys and gals! (Photograph by Bill Brown.)



**Figure 6.** Each near spacecraft's position report was put into a KML file and then opened in Google Earth in order to create this image. All the near spacecraft landed within a few miles of each other — even though the flights lasted two hours and 15 minutes — and reached altitudes between 62,000 and 73,000 feet. One reason for landing near each other is their similar ascent rates and maximum altitudes.

like to share some of that with you now. First is the flight paths taken by the four balloons I helped launch. The flight paths gradually diverge from one another as the balloons ascend, except in the case of NearSys 15O which had a much higher ascent rate than the others. It also managed to reach a higher altitude before the balloon burst.



**Figure 7.** Starting off with the airframes warm does help fight the cold. Even at the coldest part of the mission, the hand warmer chamber was 25°F warmer than the empty chamber. Next time, I'll test warming the chamber prior to flight, then removing the hand warmer just before lift-off. That will determine if the continuous production of heat is more effective or if initially heating the airframe prior to launch is more effective.

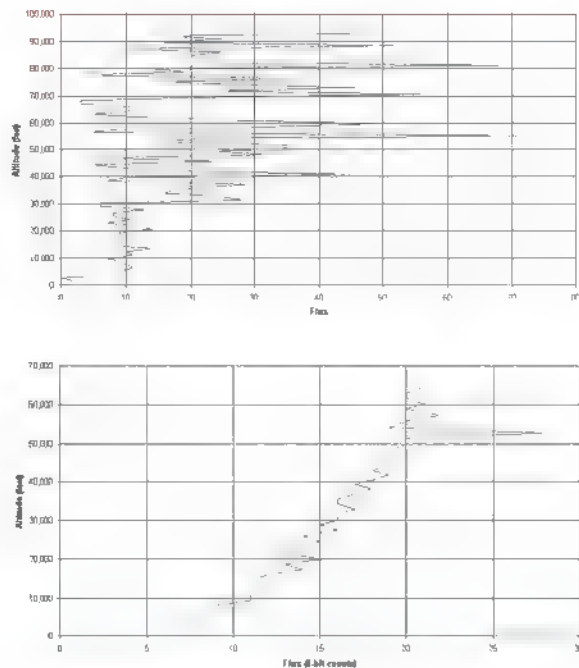
Above 60,000 feet, the winds weakened and then reversed in direction. That's most noticeable in the flight path shown for NearSys 15O.

One of the most serious threats to near space missions is the intense cold of near space. The temperature in the tropopause — or boundary between the troposphere and stratosphere — is commonly -60°F in

the summer and even colder in the winter. We construct airframes from Styrofoam™ because of its light weight, but also for its ability to insulate from this intense cold. Nevertheless, it can still get so cold inside an airframe it shuts down the electronics inside.

For example, I've seen cameras shut down due to the cold. At GPSL, I tested hand warmers as a simple lightweight method to keep electronics warm.

There are reasons to think this might not work because hand warmers operate based on the exothermic reaction of the oxidation of iron. This is a process that requires oxygen, and that's a gas in short supply as you climb into near space. However, the decrease in oxygen availability is gradual and the hand warmers have some mass that might retain some of their



**Figure 8.** Notice that the UV intensity data collected when the photometer is pointing up (NearSys 15C; top) shows a lot of variation in the data. This is because the BalloonSat was swinging and rocking during the ascent, and at times, the photometer was seeing more black sky and less sun than at other times. The atmosphere appears large to the photometer, so when it's looking down, it acts like a huge diffuser that makes the pointing direction less relevant as you can see in the NearSys 15L data. By the way, the UV-B photometer was purchased from AdaFruit (#GUVA-S12SD).

warmth during the ascent.

In addition, we can start a hand warmer an hour before the launch and get the interior of the airframe toasty warm even before lift-off.

It was my belief that warming the interior prior to launch and providing a gradually weakening heat source would help to negate the gradual cooling of the airframe's interior. The graph in **Figure 7** is from an experiment enclosed inside a BalloonSat divided into two chambers. Each chamber contained a temperature logger but only one also contained a hand warmer. The warmer was started close to an hour before launch. As you can see, it made a significant difference in that chamber's temperature.

As you may recall, I'm



experimenting with using LED-based photometers on near space missions. There's a problem with this experiment, however. The near spacecraft is not a stable platform. This means each time a photometer takes a light intensity measurement, the sun is in a different position relative to the photometer and this strongly affects the light intensity. So, on mission NearSys 15L, I tried something different turning the photometer downwards.

I didn't think this would be significant, but the data shows otherwise as you can see in **Figure 8** when you compare upward and downward photometers. Mission NearSys 15C flew the UV-B photometer facing upwards, and mission NearSys 15L flew the photometer facing downwards. It appears the downward photometer is measuring the scattering of ultraviolet off the atmosphere, which should be a good proxy for the intensity of solar ultraviolet. Better yet, the atmosphere is such a good scattering source that the pointing direction of the

In Jeff Ducklow's picture of mission NearSys 15M (**Figure 3**), the orange module is Ann Boes' tracking module which is a cube measuring one foot across in every dimension. Using that as a reference, the balloon appears to be 17 feet across. At launch, the balloon was only six feet across and about seven feet tall (for an average of 6.3 feet). That means the balloon in the image has increased its volume 19.7 times. It's pretty cold at 52,000 feet, something like -60°F. A temperature that cold will decrease the volume of the balloon by roughly 20%. Therefore, the volume of the balloon at 52,000 feet without the effects of the cold is closer to 15.8 times greater than it was at launch. That kind of expansion requires an air pressure that's 6.3% of the air pressure at the time of launch. Actual measurements made during near space missions show that the pressure is actually closer to 7.7% of surface air pressure at 52,000 feet. This back of the envelope calculation is pretty good for measuring air pressure with a photograph and using Charles' and Boyle's Laws.

photometer is irrelevant.

So, that was GPSL 2015 in a nutshell. Several things were learned and I have another idea to research for the future. If you're in the Texas area next year and interested in amateur near space exploration, then check out Granbury June 16-19. That's when ARBONET (Amateur

Radio Balloons Over North East Texas) will host the next GPSL. You can learn more about ARBONET at [www.arbonet.net](http://www.arbonet.net).

You can read about this year's GPSL host, ROBOMO at [robomo.com](http://robomo.com).

Onwards and Upwards,  
Your near space guide **NV**

# \$51<sup>For 3</sup> PCBs

FREE Layout Software!  
FREE Schematic Software!



DOWNLOAD our free CAD software

DESIGN your two or four layer PC board

SEND us your design with just a click

RECEIVE top quality boards in just days

[expresspcb.com](http://expresspcb.com)

# ELECTRONET

**CORIDIUM**  
 Floating point **BASIC** for  
**ARM** controllers from \$5.00  
[www.coridium.us](http://www.coridium.us)

**ALL ELECTRONICS**  
 Electronic Parts and Supplies.  
[www.allectronics.com](http://www.allectronics.com)  
 Free 96 page catalog 1-800-828-5432

For the ElectroNet  
 online, go to  
[www.nutsvolts.com](http://www.nutsvolts.com)  
 click **Electro-Net**

**USB** Add USB to your next project--  
 It's easier than you might think!  
**DLP Design**  
 USB-FIFO • USB-UART • USB/Microcontroller Boards  
 RFID Readers • Design/Manufacturing Services Available  
*Absolutely NO driver software development required!*  
[www.dlpdesign.com](http://www.dlpdesign.com)

**INVEST in your BOT!**



**INVEST in HITEC**

12115 Payne Street Poway, CA 92064 858-748-6948 [www.hitecrobot.com](http://www.hitecrobot.com)

**HOBBY ENGINEERING**  
 Kits, Parts and Supplies  
[www.HobbyEngineering.com](http://www.HobbyEngineering.com)

PCB, PCBA and More! **Myro** Low cost High Quality  
[www.myropcb.com](http://www.myropcb.com)  
 1-888-PCB-MYRO

**Ironwood ELECTRONICS** GHz BGA/QFN Sockets  
 0.3mm to 1.27mm  
 Industry's Smallest Footprint  
 1-800-410-1020  
[www.ironwoodelectronics.com](http://www.ironwoodelectronics.com)



**superbrightleds.com**  
 COMPONENT LEDs • LED BULBS • LED ACCENT LIGHTS



The fun begins where the road ends! **WARDEN** ServoCity.com  
 Only \$199.99



**ramsey** GET THE NUTS&VOLTS DISCOUNT!  
 Mention or enter coupon code **NVRMZ142**  
 and receive 10% off your order!  
[www.ramseyvolts.com](http://www.ramseyvolts.com)  
 AM/FM Broadcasters • Hobby Kits  
 Learning Kits Test Equipment  
 ...AND LOTS OF NEAT STUFF!



**MOBILE APP NOW AVAILABLE!**

**SERVO**  
 Download NOW On Your Favorite Mobile Device!

**FOR ROBOT BUILDERS**

**IOS • ANDROID • KINDLE FIRE**

[www.servomagazine.com](http://www.servomagazine.com)

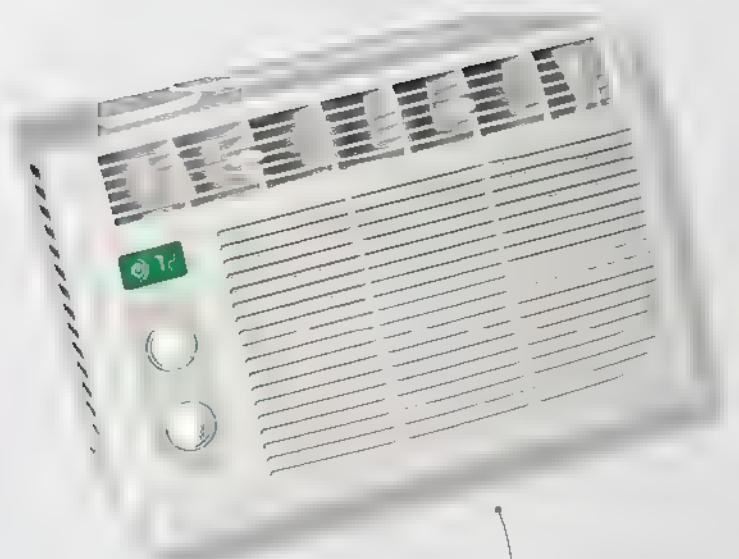
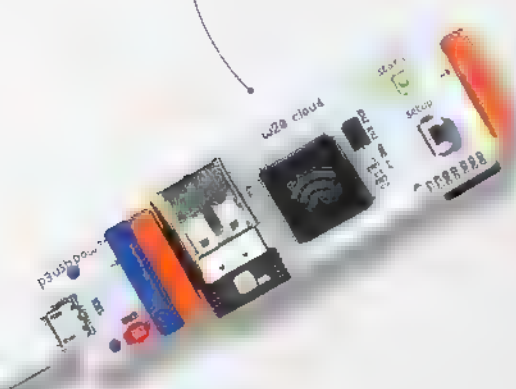


Did you know that **Nuts & Volts** now has a weekly content newsletter? **Want to get it?**  
 You have three ways to sign up:

- Visit us on Facebook and click on "Join My List."
- Using your cell phone, send "NVNEWSLETTER" as a text message to 22828.
- Visit the **Nuts & Volts** FAQs to sign up at [nutsvolts.com/faqs](http://nutsvolts.com/faqs)

# littleBits™

*Connect to the internet!*



*way bigger in real life*

## PROTOTYPE SMART DEVICES IN MINUTES.

littleBits makes a platform of easy-to-use electronic building blocks, enabling you to prototype your ideas faster. And smarter. Learn more at [LITTLEBITS.CC](http://LITTLEBITS.CC)



# From Data Logger to On-Demand Data Storage Device

Despite the “works out of the box” nature of the open source-based OpenLog data logger, you still have to write some code to access its resources. It really doesn’t matter if you purchase an OpenLog from SparkFun or build your own custom version. You will still need to do some coding.

The idea behind the OpenLog is instant data logging via a high speed serial port. With some clever coding, the OpenLog can also be used as a microSD-based disk drive. My mission this month is to take the pain out of using the OpenLog as an on-demand data storage device. We already have some custom Open\_log hardware. So, let’s get crackin’ with that code.

## The Programming Model

As you can see in **Schematic 1**, the OpenLog circuit consists of an Atmel ATmega328P microcontroller, a couple of LEDs, a 16 MHz resonator, a microSD card socket, and a minimal complement of resistors and capacitors. The OpenLog firmware doesn’t check for the presence of a microSD card. So, the FDN339AN MOSFET

and its associated resistors in my adaptation are really not necessary for proper operation of the OpenLog.

Since all of the microSD SPI interface coding is handled by the OpenLog firmware, our code only needs to concentrate on the OpenLog’s serial interface. Indication of proper data transfer on both the SPI and serial interfaces has already been taken care of, as well. The LED attached to the ATmega328P’s PD5 pin responds to signals flowing on the serial port. The SPI signal transitions can be observed by visually monitoring the LED attached to pin PB5.

The latest version of the OpenLog firmware includes a bootloader that allows programming of the ATmega328P via its serial interface. I chose not to employ the services of the bootloader and instead use a standard AVR programmer to load the OpenLog firmware.

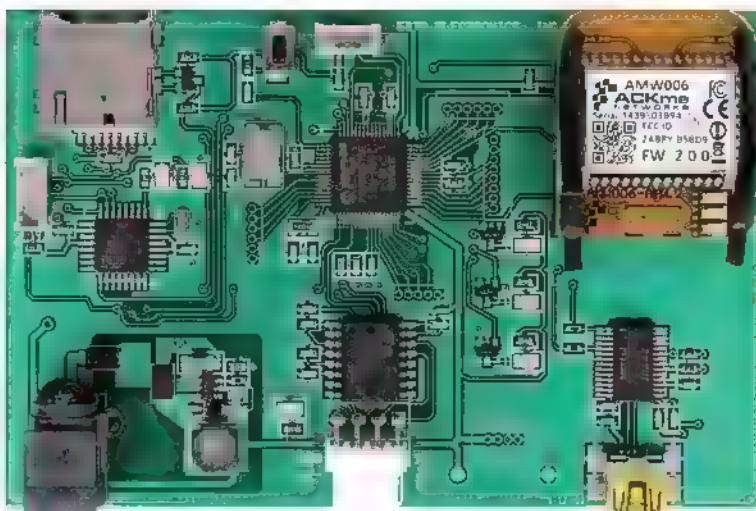
We will use the PIC32MX575F512H hardware I recently introduced to you in a previous edition of Design Cycle. The OpenLog library we are about to create will also drive the PIC32MX795F512H board, which was also featured in a recent Design Cycle column. The 575F hardware is rendered in **Photo 1**.

The OpenLog library routines will be written using the Microchip XC32 C compiler. The resultant C source code can be easily ported to Arduino, CCS C, or PICBASIC Pro.

## Built-in Tools

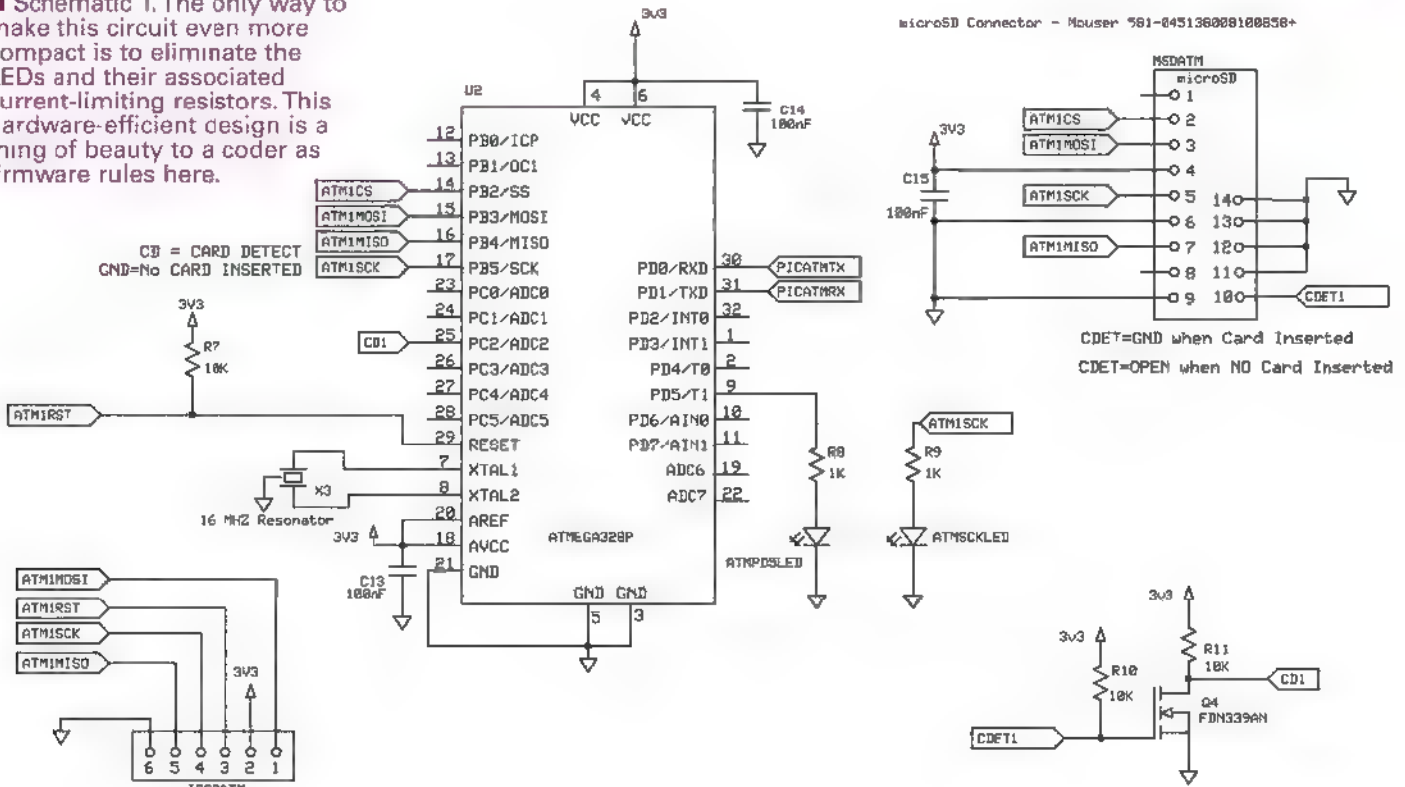
We must have a working knowledge of how the OpenLog firmware responds to the commands that we will issue. Ideally, we need an OpenLog device that is attached to a PC serial port. We would then be able to use a terminal emulator to monitor the OpenLog commands and responses. The good news is that we don’t need a separate OpenLog module.

We can use our custom OpenLog implementation and the onboard FTDI USB IC. All



■ Photo 1. We are primarily interested in the upper lefthand corner of the board. An added plus is the FTDI USB portal you see in the lower righthand corner. With a bit of clever coding, the OpenLog serial interface can be attached logically to the FTDI FT232RL via the PIC32MX575F512H. This allows the OpenLog to be exercised by a PC serial port.

■ **Schematic 1.** The only way to make this circuit even more compact is to eliminate the LEDs and their associated current-limiting resistors. This hardware-efficient design is a thing of beauty to a coder as firmware rules here.



we need is a little bit of code:

```

//*****
// MAIN
//*****
int main (void)
{
    BYTE bitein;
    init();
    //*****
    // TERMINAL INTERFACE
    //*****
    do{
        if(CharInQueue1())
        {
            do{
                bitein  recvchar1();
                sendBiteUart2(bitein);
            }while(CharInQueue1());
        }
        if(CharInQueue2())
        {
            do{
                bitein  recvchar2();
                sendBiteUart1(bitein);
            }while(CharInQueue2());
        }
    }while(1);
}

```

All of the magic is performed within the *do-while* loop. The *CharInQueue1* function alerts us that a character has been received from the FTDI USB IC. The *CharInQueue2* function performs the identical alert notification for the ATmega328P. When a character is received from the PC via the FTDI USB interface — which

is depicted in **Schematic 2** — we want to forward that character to the ATmega328P's receive (PD0/RXD) pin.

To establish a bidirectional data path, we also need to send any characters emanating from the ATmega328P to the FT232RL's receive pin. The USB-to-Atmel pipe is realized in the *CharInQueue1* portion of the code and the Atmel-to-USB pipe is laid by the *CharInQueue2* portion of the code.

At power-up or following a RESET command, the OpenLog firmware will post an informative prompt, which reads *12>*. The 1 tells us that the serial connection has been established. The 2 indicates that the microSD card successfully initialized. In our case, the > character tells us that the OpenLog is in command mode. As we're going to be issuing commands to the OpenLog via firmware, I modified the CONFIG.TXT file to force the OpenLog firmware to come up in command mode. Otherwise, the prompt would read *12<*, which indicates that the OpenLog is ready to receive a serial stream of data to be logged.

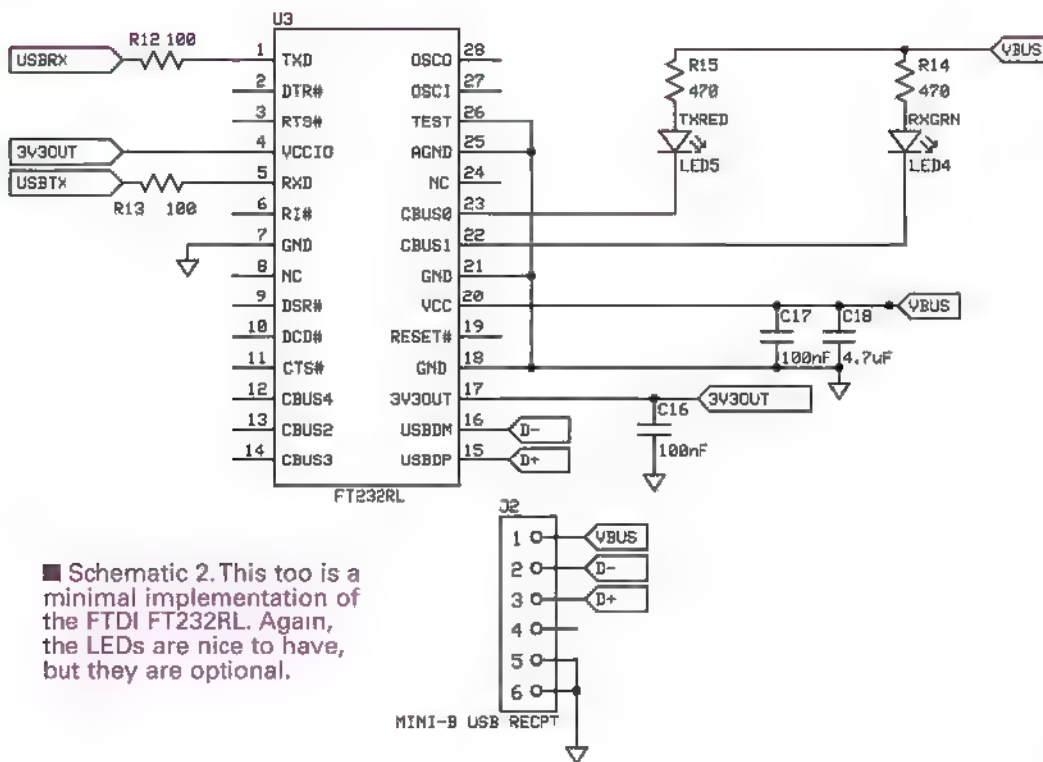
Once the OpenLog firmware is allowed to run, the OpenLog will automatically create a CONFIG.TXT file on a blank formatted microSD card. Here's a peek at the modified CONFIG.TXT file's contents:

```

9600,26,3,2,1,1,0
baud,escape,esc#,mode,verb,echo,ignoreRX

```

The text follows the numeric characters in the order in which they appear. OpenLog's default baud rate is 9600



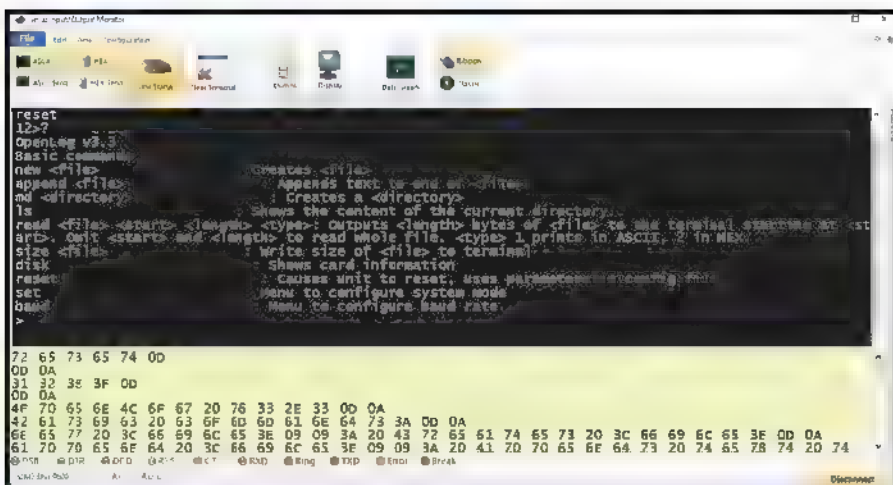
■ Schematic 2. This too is a minimal implementation of the FTDI FT232RL. Again, the LEDs are nice to have, but they are optional.

bps. The escape character is 0x26, which is issued by a keyboard as Cntl+z and it takes three escape characters to drop into command mode from logging mode. My modification to the CONFIG.TXT file contents is represented by the 2, which tells the OpenLog firmware to come up in command mode instead of logging mode.

Since we want to manipulate the microSD card from a PC using the onboard FTDI USB portal, I left the *verbose* and *echo* settings at their defaults. I've already done all of the necessary preliminary work of defining the I/O pins and activating the necessary PIC32MX575F512H

RESET command, which should return a prompt. As you can see in **Screenshot 1**, we've had a good start. Everything sent and received from the OpenLog is visible in ASCII and hexadecimal form. Once I received a prompt, I entered a ? character to reveal the available commands.

The goal of having this view is to note the characters that surround the data. When we go "microcontroller automatic," we will need to filter out the characters that are not part of the data we wish to retrieve. For example, note the carriage return/line feed characters before the 0x3E prompt in **Screenshot 2**. Keep that in mind as you examine the code we're about to write.



■ Screenshot 1. Things are good. The OpenLog firmware is communicating with the Serial Input/Output Monitor application running on my Lenovo laptop by way of the PIC32MX575F512H.

peripherals. So, let's fire up the electronics that support the OpenLog and see if the code works.

## Manual Mode

While we are operating the OpenLog in manual mode, we will use the Serial Input/Output Monitor application that comes standard with the CCS C Compiler to issue commands and capture the results.

Our OpenLog firmware will come up in command mode. Our PIC32MX575F512H firmware swallows up the initial prompt to make sure the OpenLog hardware is ready to rock. So, the first command I will issue is the

## Initializing the microSD Hardware

The OpenLog was originally designed to be an Arduino accessory. In our case, the OpenLog hardware is under the control of a PIC32MX575F512H, which has access to the ATmega's RESET pin. Before attempting to issue commands via firmware, the PIC32MX575F512H needs to know if the OpenLog hardware is functional and if a functional microSD card is mounted in the OpenLog's socket. There is one sure fire way to get the ATmega's attention

and get the initialization information the PIC32MX575F512H desires. Force the ATmega328P to perform a hardware reset:

```
//Enable Interrupts
INTConfigureSystem(INT_SYSTEM_CONFIG_MULT_VECTOR);
ei();
tdelaysms(10);
atmrstLO;
bluLED On;
redLED On;
tdelaysms(100);
atmrstHI;
```

Our PIC32MX575F512H firmware uses the onboard blue and red LEDs as microSD status indicators. Since all of the UARTs utilize receive interrupts, we must enable interrupts before we force the ATmega328P into reset. As you can see, we turn both the blue and red LEDs on just after we send the ATmega328P to reset land. One hundred milliseconds later, we release the ATmega328P's reset pin. We will give the OpenLog firmware a maximum of five seconds (5,000 ms) to respond following the release of the ATmega328P's reset pin. We're looking for the 12> prompt to be sent by the ATmega328P:

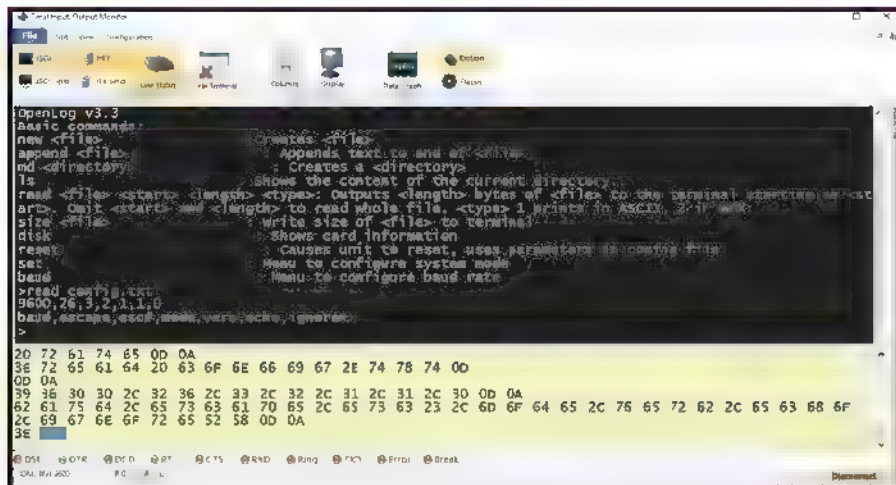
```
const BYTE atmelPrompt[] "12>";

i 0;
timeoutms 5000;
do{
  if(CharInQueue2())
  {
    do{
      rxBufAtmel[i++] = rcvchar2();
    }while CharInQueue2();
    tdelaysms(1);
  }
}while(!timeoutms && strcmp(atmelPrompt, rxBufAtmel) != 0);
```

If the ATmega328P responds within the time limit we specified, we put on a light show with the blue and red LEDs:

```
if(timeoutms)
{
  for(i 0;i<15;++i)
  {
    bluLED Tog;
    tdelaysms(25);
    redLED Tog;
    tdelaysms(25);
  }
  bluLED Off;
  redLED Off;
}
```

If the serial connection is down and/or the microSD card is not detected, we are terminal. In the case of an initialization error, the blue and red



■ Screenshot 2. These are the available commands. I've issued the read command here with no options. What we're really looking for are the characters that surround the data.

LEDs are flashed in unison continually:

```
else
{
  di();
  bluLED Off;
  redLED Off;
  do{
    bluLED Tog;
    redLED Tog;
    tdelaysms(100);
  }while(1);
}
```

The only way to restart from an initialization error is rectify the error situation and restart the PIC32MX575F512H; 99.9999% of the time, the response timeout and initialization error condition are caused by an empty microSD socket.

## Microcontroller Mode

Now that we've put the PIC32MX575F512H in control, let's take it a step further and write a function to create a file on the microSD card. However, before we do that, let's turn off the verbose and echo modes in our CONFIG.TXT file:

```
9600,26,3,2,0,0,0
baud,escape,esc#,mode,verb,echo,ignoreRX
```

Now, let's do some coding:

```
/**
 * *****
 ** UART2 TX String
 * *****
 */
void sendStrUart2(const char* buffer)
{
  // transmit till NULL character is
  // encountered
  while(*buffer != '\0')
  {
    (U2STAbits.TRMT = 0);
  }
}
```

```

    U2TXREG (*buffer++);
}
//*****
/* CREATE NEW FILE
//*****
void newFile(BYTE filename)
{
    BYTE bitein;
    char cmdBuf[32];

    //send new file command
    sprintf(cmdBuf, "new
microsd%d.csv\r", filename);
    sendStrUart2(cmdBuf);
    //wait for command mode prompt
    do{
        if(CharInQueue2())
        {
            bitein  recvchar2();
        }
    }while(bitein != '>');
}

```

The maximum length of our filenames is eight characters plus three characters behind the dot delineator (12345678.123). So, we can create up to 10 files with our filename of *microsdX.csv*, where X represents the *filename* argument that can range from 0 through 9. The *.csv* extension allows us to read the file in a formatted manner using Microsoft Excel. The *new* command is issued from our PIC32MX575F512H firmware just as it would be manually (reference **Screenshot 1**).

The *sprintf* function writes the *new* command string into the *cmdBuf* array and appends a null character (0x00) to the command string. The null character is important as our *sendStrUart2* function is looking for a null character to indicate the end of the string. Once the *new* command is transmitted to the OpenLog firmware, we wait for a > prompt following the completion of the command.

We can instantly verify that the file was created by utilizing the onboard FTDI USB portal. All we need is a simple piece of code targeting the *ls* command:

```

//*****
/* LIST File
//*****
void listFile(void)
{
    char cmdBuf[32];
    WORD timeouts;
    WORD i;
    BYTE bitein;

    //send list file command
    sprintf(cmdBuf, "ls\r");
    sendStrUart2(cmdBuf);
    //wait for card to respond to ls
    i 0;
    timeouts 5000;
    do{
        if(CharInQueue2())
        {
            do{
                bitein  recvchar2();
                sendBiteUart1(bitein);
            }while(CharInQueue2());

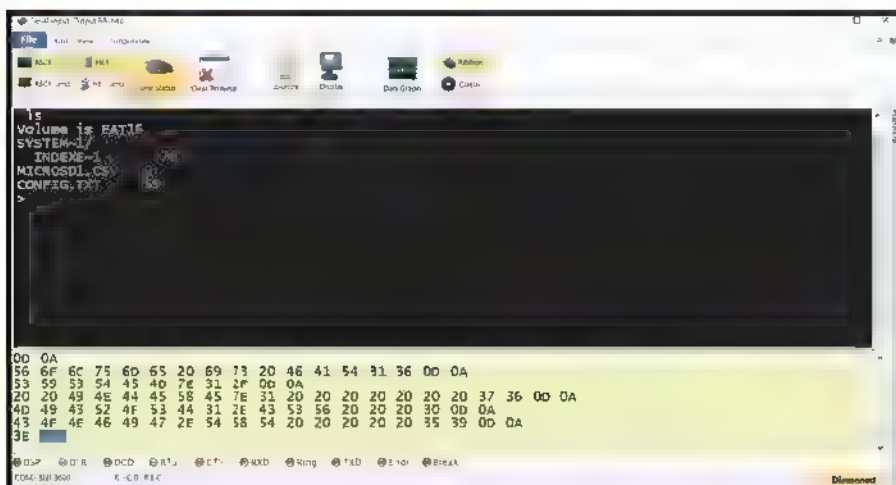
            tdelaysms(1);
        }while(timeouts-- && (bitein != '>'));
    }
}

```

Again, we use the *sprintf* function to load and format our command string. And again, we send the *ls* command string using the *sendStrUart2* function. If the OpenLog firmware doesn't return a result in five seconds, we've got big problems to solve and probably would not have even gotten this far in our coding. So, we'll wait for the file listing to flow to the PIC32MX575F512H's UART2. Every character that is received by UART2 is forwarded to the transmit pin of UART1 (the FTDI USB portal). As you can see in **Screenshot 3**, everything is okay. There are currently two files on the microSD card we can list and both of them show up in our **Screenshot 3** listing.

## Write to Our New File

We have successfully created a file called



**Screenshot 3.** This is exactly what we should have expected. The *CONFIG.TXT* file — which was created by the OpenLog firmware — and a newly created and empty *MICROSD1.CSV* file.

Original OpenLog Module  
SparkFun  
[www.sparkfun.com](http://www.sparkfun.com)

PIC32MX575F512H  
XC32 C Compiler  
Microchip  
[www.microchip.com](http://www.microchip.com)

Serial Input/Output Monitor  
CCS  
[www.ccsinfo.com](http://www.ccsinfo.com)

ATmega328P  
Atmel  
[www.atmel.com](http://www.atmel.com)

PICBASIC Pro  
M.E. Labs  
[www.PBP3.com](http://www.PBP3.com)



microsd1.csv. However, according to the file listing we see in **Screenshot 3**, the new file is empty. No worries. We've got code for that:

```

//*****
//* WRITE File
//*****
void writeFile(BYTE filename)
{
    char cmdBuf[32];
    BYTE bitein;

    //send write command
    sprintf(cmdBuf,"write
microsd%d.csv
0\r",filename);
    sendStrUart2(cmdBuf);
    //wait for card to indicate
    //file is open
    //and ready for writing
    do{
        if(CharInQueue2())
        {
            bitein  recvchar2();
        }
        }while(bitein != '<');

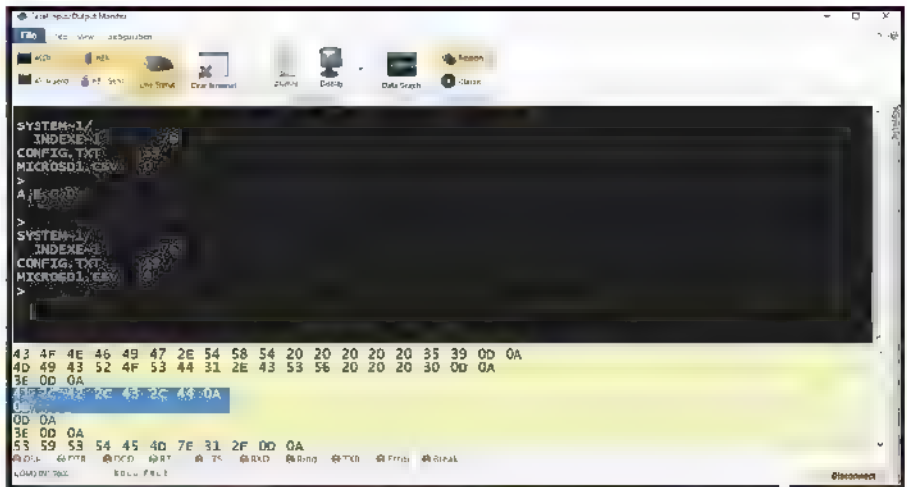
    //put data to write here
    sendBiteUart2('A');
    sendBiteUart2(',');
    sendBiteUart2('B');
    sendBiteUart2(',');
    sendBiteUart2('C');
    sendBiteUart2(',');
    sendBiteUart2('D');
    //commit the data to the card
    sendBiteUart2(0x0D);
    //send an empty line to end the write
    sendBiteUart2(0x0D);
    //wait for command mode prompt
    do{
        if(CharInQueue2())
        {
            bitein  recvchar2();
        }
        }while(bitein != '>');
}

```

We can tell the OpenLog firmware where to begin our *write* operation. I specified a start position of 0 (zero) in the *write* command string. The preparation and sending of the *write* command is no different than what we have seen thus far. Once the *write* command has been issued, we wait for the OpenLog firmware to signal that it is ready to write our data to the microSD card with a < character. Once we sense the < character, we can start to feed our data into the microSD card using a series of *sendBiteUart2* function calls.

If we were sending strings of data, we could use the *sendStrUart2* function. Note that commas delimit the data being sent to the microSD card. The commas are used by Excel to place each byte of data in a separate cell.

When all of the data has been transmitted, the OpenLog firmware wants to see an 0x0D (carriage return) character, which signals the end of the data packet transmission. The *write* operation is complete when we receive a > prompt character from the OpenLog firmware.



■ **Screenshot 4.** I included a couple of list file commands in this sequence. This spin of the *read* function works just fine as long as all of the data is printable ASCII characters. After the data is written, the MICROSD1.CSV file size is nine bytes not seven. The highlighted hex dump shows the extra two characters as a carriage return (0x0D) and a line feed (0x0A).

## Let's Read It

This version of a *read* function will return readable ASCII characters only. If the character falls outside of the printable character realm, a period (0x2E) will be displayed instead:

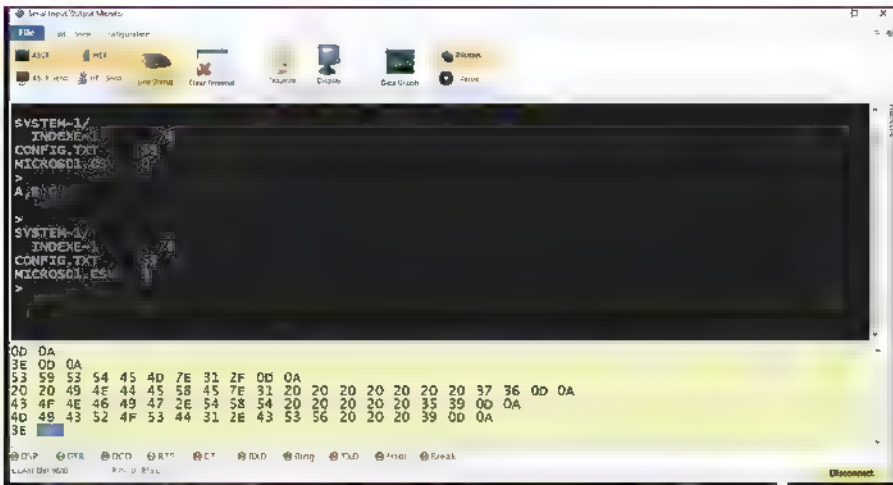
```

//*****
//* READ File
//*****
void readFile(BYTE filename)
{
    char cmdBuf[32];
    WORD timeouts;
    WORD i;
    BYTE bitein;

    memset(rxBufAtmel,0x00,sizeof(rxBufAtmel));
    //send read command
    sprintf(cmdBuf,"read
microsd%d.csv\r",filename);
    sendStrUart2(cmdBuf);
    //wait for card to respond to read -
    //about 18mS
    i 0;
    timeouts 5000;
    do{
        if(CharInQueue2())
        {
            bitein  recvchar2();
            sendBiteUart1(bitein);
            rxBufAtmel[i++]  bitein;
        }
        }while(timeouts-- && (rxBufAtmel[i-1] != '>'));
}

```

The *read* function algorithm is no different than any of the other command functions we've written so far. The data read from the microSD card is stored in a buffer

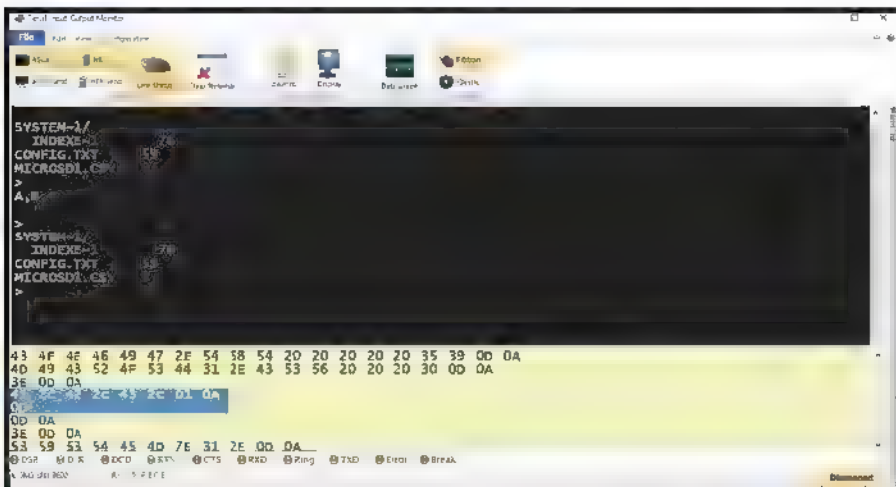


■ Screenshot 5. I substituted a hexadecimal 1 (0x01) for the D character in our A,B,C,D data packet. As you can see, a period printed as 0x01 is a control character and not a printable character.

(*rxBufAtmel*) after it is sent to the FTDI USB portal.

Check out **Screenshot 4**. Our MICROSD1.CSV is initially empty. Then, we write seven bytes to the file (A,B,C,D). The next operation is a *read* and we see those results printed in **Screenshot 4**. Finally, another list file command is executed and we see the MICROSD1.CSV file size is nine bytes. If you take a look at the highlighted hex dump in **Screenshot 4**, you will see that a carriage return character and a line feed character have been appended to the file data by the OpenLog firmware.

This is all great if we only need to store printable ASCII characters. **Screenshot 5** shows what happens when a hexadecimal 1 (0x01) is substituted for the D character in our A,B,C,D data packet. The OpenLog firmware prints a period (0x2E) for nonprintable ASCII characters. This



■ Screenshot 6. Now *that* is a REAL *read* function. We can now store and recall ASCII and hexadecimal characters. Note that the hexadecimal 1 is represented correctly in the highlighted hex dump area of this shot.

means we can't store and recall hexadecimal numbers from the microSD card correctly. So, how do we code a *read* function that allows us to store ASCII characters and hexadecimal values? Like this:

```

/*****
/** READ File
*****/
void readFile(BYTE fileNum)
{
    char cmdBuf[32];
    WORD timeoutms;
    WORD i;
    WORD fileSize;
    BYTE bitein;

    fileSize = getFileSize(
        fileNum);

```

```

    memset (rxBufProg, 0x00,
        sizeof (rxBufProg));
    //send read command

    sprintf (cmdBuf, "read
microsd%d.csv 0 %u
3\r", fileNum, fileSize);
    sendStrUart2 (cmdBuf);
    //wait for card to respond to
    //read - about 18mS
    i = 0;
    timeoutms = 5000;
    do {
        if (CharInQueue2 ())
            bitein = rcvchar2 ();
            sendBiteUart1 (bitein);
            rxBufProg [i++] = bitein;
        }
        tdelays (1);
    } while (timeoutms-- && (rxBufProg [i-1]
        != '\r'));
}

```

The file read results in **Screenshot 6** verify our updated *readFile* function. The *read* command syntax has drastically changed here. We have instructed the OpenLog firmware to read from file position 0 in raw mode.

To be able to enter the raw argument (3), we must provide a file size. And yes, there's code for that:

```

/*****
/** GET FILE SIZE
*****/
WORD getFileSize (BYTE fileNum)
{
    char cmdBuf [32];
    WORD timeoutms;
    WORD i;
    WORD fileSize;
    char sizeBuf [8];

    memset (sizeBuf, 0x00,
        sizeof (sizeBuf));

    //send size command

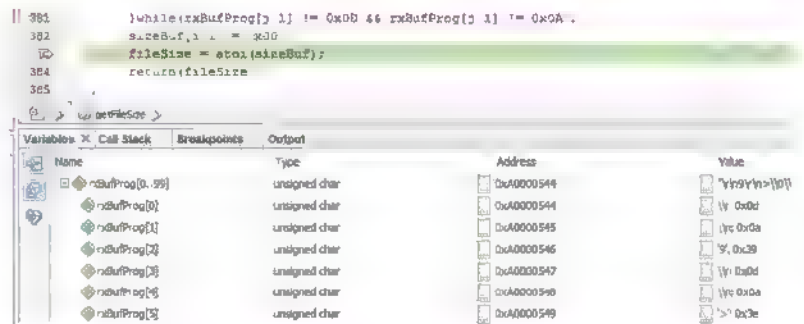
```

```

sprintf(cmdBuf, "size
microsd%d.csv\r", fileNum);
sendStrUart2(cmdBuf);
//wait for card to respond to
//read - about 18ms
i 0;
timeoutms 5000;
do{
    if(CharInQueue2())
    {
        rxBufProg[i++]
        recvchar2();
    }
    tdelaysms(1);
}while(timeoutms- && (rxBufProg
[1-1]
! '>'));
i 0;
j 0;
do{
    if(isspace(rxBufProg[j]))
    {
        ++j;
    }
}while(isspace(rxBufProg[j]));

do{
    sizeBuf[i++] rxBufProg[j++];
}while(rxBufProg[j-1] != 0x0D &&
rxBufProg[j-1] != 0x0A);
sizeBuf[1-1] 0x00;
fileSize atoi(sizeBuf);
return(fileSize);
}

```



■ Screenshot 7. I have set a breakpoint just before the `atoi` function is called. In the Variables window, you can see how the raw file size information is returned by the OpenLog firmware.

```

do{
    bluLED Tog;
    tdelaysms(250);
}while(1);

```

I had a blast writing this code. I hope you will find it useful. You can get the complete source library via download from the article link. **NV**

File size is returned as a set of ASCII characters surrounded by carriage return/line feed combinations. Carriage returns and line feeds are considered white space characters. The `getFileSize` function wades through the white space characters using the `isspace` function until an ASCII character is encountered. The ASCII characters that make up the file size data are moved to `sizeBuf` and a null character is appended.

The file size is obtained by performing an ASCII to Integer conversion using the `atoi` function. In **Screenshot 7**, I set a breakpoint prior to the `atoi` function call to capture the raw buffered file size data returned by the OpenLog firmware.

## Exit Function

Yep. There's code for that. The resultant `main` function we used to read and write to the microSD card looks like this:

```

//*****
// MAIN
//*****
int main (void)
{
    BYTE bitein;
init();
    newFile(1);
    listFile();
    writeFile(1);
    readFile(1);
    listFile();
}

```

The advertisement for RF Specialists features several product categories:

- FCC Part 90 Compliant:** USX2 - NBPM Multi-Channel UHF Transceiver with Programmable RF Power.
- MURS (Multi-Use Radio Service):** SHX1 - Long Range, High Power MURS Band Transceiver.
- ZigBee Pro:** OEM Modules and USB ZigBee sticks. Mesh Networks.
- Industrial Bluetooth:** OEM, Modules, Wireless Device Servers, RS-232. Long range options, low cost.
- Ultra-Low Power Wi-Fi networking module and Eval board:** The AM7006 'Numbat' module is an ultra-low power Wi-Fi networking module with full regulatory certification.
- RF Design Services:** Prepared to work with your in-house engineers, or support your RF project from initial design to implementation. Applications include Medical, Smart Grid Meters, SCADA, Lighting Control.

**LE MOS INTERNATIONAL** Tel: 1.866.345.3667 orders@lemosint.com www.lemosint.com

# The Nuts & Volts **WEBSTORE**

## GREAT FOR DIYers!

### Beginner's Guide to 3D Printing by Chuck Hellebuyck

This book was written to answer the questions most beginners need answered. It covers many of the popular 3D printer choices and then uses the under \$500 Da Vinci 1.0 from XYZprinting to show how easy it is to get started. Chuck takes you through using Tinkercad software for creating your own custom designs, then goes further and shows you how to take a simple design and send it off to a professional 3D printer. This book was designed for anyone just getting started. **\$24.95**



### Electronics Workshop Companion for Hobbyists by Stan Gibilisco

In this practical guide, electronics expert Stan Gibilisco shows you step-by-step how to set up a home workshop so you can invent, design, build, test, and repair electronic circuits and gadgets. *Electronics Workshop Companion for Hobbyists* provides tips for constructing your workbench and stocking it with the tools, components, and test equipment you'll need. Clear illustrations and interesting do-it-yourself experiments are included throughout this hands-on resource. **\$25.00**



### Arduino Projects for Amateur Radio

by Jack Purdum, Dennis Kidder  
**Boost Your Ham Radio's Capabilities Using Low Cost Arduino Microcontroller Boards**

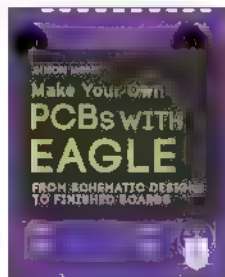
Do you want to increase the functionality and value of your ham radio without spending a lot of money? This book will show you how! *Arduino Projects for Amateur Radio* is filled with step-by-step microcontroller projects you can accomplish on your own — no programming experience necessary.



Reg Price \$30.00 Sale Price \$24.00

### Make Your Own PCBs with EAGLE by Eric Kleinert

Featuring detailed illustrations and step-by-step instructions, *Make Your Own PCBs with EAGLE* leads you through the process of designing a schematic and transforming it into a PCB layout. You'll then move on to fabrication via the generation of standard Gerber files for submission to a PCB manufacturing service. This practical guide offers an accessible, logical way to learn EAGLE and start producing PCBs as quickly as possible.



**\$30.00**

### Build Your Own Transistor Radios by Ronald Quan

#### A Hobbyist's Guide to High Performance and Low-Powered Radio Circuits

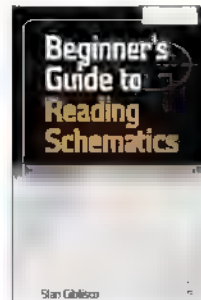
Create sophisticated transistor radios that are inexpensive yet highly efficient. Inside this book, it offers complete projects with detailed schematics and insights on how the radios were designed. Learn how to choose components, construct the different types of radios, and troubleshoot your work. **\*Paperback, 496 pages**



Reg Price \$49.95 Sale Price \$39.95

### Beginner's Guide to Reading Schematics, 3E by Stan Gibilisco

Navigate the roadmaps of simple electronic circuits and complex systems with help from an experienced engineer. With all-new art and demo circuits you can build, this hands-on, illustrated guide explains how to understand and create high-precision electronics diagrams. Find out how to identify parts and connections, decipher element ratings, and apply diagram-based information in your own projects.

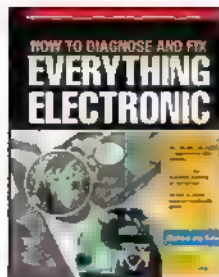


**\$25.00**

### How to Diagnose and Fix Everything Electronic by Michael Jay Geier

#### Master the Art of Electronics Repair

In this hands-on guide, a lifelong electronics repair guru shares his tested techniques and invaluable insights. *How to Diagnose and Fix Everything Electronic* shows you how to repair and extend the life of all kinds of solid-state devices, from modern digital gadgetry to cherished analog products of yesteryear. **\$24.95**



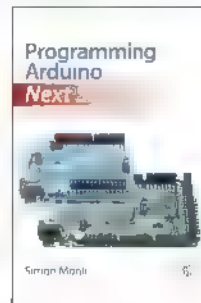
### Programming PICs in Basic by Chuck Hellebuyck

If you wanted to learn how to program microcontrollers, then you've found the right book! Microchip PIC microcontrollers are being designed into electronics throughout the world and none is more popular than the eight-pin version. Now the home hobbyist can create projects with these little microcontrollers using a low cost development tool called the CHIPAXE system and the Basic software language. Chuck Hellebuyck introduces how to use this development setup to build useful projects with an eight-pin PIC12F683 microcontroller. **\$14.95**



### Programming Arduino Next Steps: Going Further with Sketches by Simon Monk

In this practical guide, electronics guru Simon Monk takes you under the hood of Arduino and reveals professional programming secrets. Also shows you how to use interrupts, manage memory, program for the Internet, maximize serial communications, perform digital signal processing, and much more. All of the 75+ example sketches featured in the book are available for download. **\$20.00**



Order online @ [store.nutsvolts.com](http://store.nutsvolts.com)  
 Or CALL 1-800-783-4624 today!

**EDUCATIONAL**

**Beginners Guide Book Combo.**

**Only \$85.95**  
 Plus  
 FREE Priority Mail Shipping  
 US Only

**DO YOU LOVE RADIOS?**

**Sale Price \$39.95**      **Sale Price \$19.95**      **Sale Price \$23.95**

**Get 20% off these three books**  
 To order call 800 783-4624 or visit: <http://store.nutsvolts.com>

Arduino Classroom - learn computing and electronics

The free Internet virtual textbook: Arduino 101 at [www.arduinoclassroom.com](http://www.arduinoclassroom.com) provides a sensible learning sequence that introduces computing and electronics with clear text and detailed hands-on labs with tested examples using the Arduino Projects Kit.

Available from Nuts&Volts for only \$44.99

**CD-ROM SPECIAL**

**Nuts & Volts  
 11 CD-ROMs  
 & Hat Special!**  
 That's 132 issues!  
 Complete with supporting  
 code and media files.

**Only \$229.95**  
 or \$24.95 each.

**The Nuts & Volts  
 Pocket Ref**  
 All the info you need at your fingertips!

This great little book is a concise all-purpose reference featuring hundreds of tables, maps, formulas, constants & conversions.

**Only \$12.95** AND it all fits in your shirt pocket!

Visit <http://store.nutsvolts.com> or call (800) 783-4624

## PROJECTS

### Fading Eyes Deluxe Board



The fading eyes circuit gives you two LED eyes that can be adjusted between a slow fade-in/fade-out to quick pulses. The speed is changed by simply adjusting the variable resistor with a small screw driver. Another adjustment allows you to set how long the LED stays on within the on/off fade cycle. Includes a battery snap and two red LEDs and two 24 inch eye cables.

**\$16.95**

### Talking Skull Kit



*It's back!*

The new and improved Talking Skull Kit. This latest version includes an improved audio and servo driver board. Go to our webstore online to see all the improvements firsthand.

This kit provides everything necessary to build one talking skull. You provide the labor and tools, and you'll have a great Halloween prop in no time!

**\$97.95**

### Peek-a-Boo Ghost Kit



The Peek-a-Boo Ghost kit is a fun, low cost multi-use microcontroller kit. When triggered by the included motion sensor, this mini animatronic waves its arms, lights its LED eyes, and plays back the sounds you record. Perfect for kids, this kit can be used to create a fun Halloween prop for your desk, front door, or walkway. Watch the video to see this cool kit in action. Available in both a program-it-yourself or with a pre-programmed PICAXE chip option.

Un-programmed Chip Kit **\$29.95**

Pre-programmed Chip Kit **\$37.95**

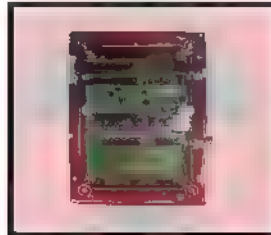
### Solar Charge Controller Kit 2.0



If you charge batteries using solar panels, then you can't afford not to have them protected from over-charging. This 12 volt/12 amp charge controller is great protection for the money. It is simple to build, ideal for the novice, and no special tools are needed other than a soldering iron and a 9/64" drill!

**\$27.95**

### Geiger Counter Kit



This kit is a great project for high school and university students. The unit detects and displays levels of radiation, and can detect and display dosage levels as low as one micro-roentgen/hr. The LND 712 tube in our kit is capable of measuring alpha, beta, and gamma particles.

*Partial kits also available.*

**\$159.95**

### 3D LED Cube Kit



This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes.

Colors available: Green, Red, Yellow & Blue

**\$57.95**

## FOR BEGINNER GEEKS!

### The Learning Lab 1 Fundamental Concepts



**\$59.95**

### The Learning Lab 2 Basic Digital Concepts and Op-Amps



**\$49.95**

### The Learning Lab 3 Basic Electronics, Oscillators and Amplifiers



**\$39.95**

These labs from LF Components show simple and interesting experiments and lessons, all done on a solderless circuit board.

As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

**For more info and lab details, please visit our webstore.**

## NEW PRODUCTS *Continued from page 21*

and error frames. The decodes use a color-coded overlay that clearly identifies different parts of the data being captured, allowing the user to quickly identify different parts of the CAN FD and FlexRay data such as the frame IDs, status bits, and message data

CAN FD is the next generation of the popular CAN standard. As CAN has reached the bandwidth limits of what it can transmit, CAN FD enables higher bit rates and longer data payloads. CAN FD goes beyond the 1 Mb/s limit of classic CAN and will transmit data at up to 10 Mb/s. CAN FD data payloads may now consist of 64 bytes per frame as opposed to the eight-byte per frame limit of classic CAN.

FlexRay is an automotive communication standard that was first used in vehicles in 2006. FlexRay supports data rates up to 10 Mb/s, static and dynamic transmission, synchronous and asynchronous message protocols, and has a time triggered deterministic architecture.

Additionally, FlexRay can have two independent channels for data communications which allows for redundancy and fault-tolerance. The WS3K-CAN FDbus TD and WS3K-FlexRaybus TD packages for the WaveSurfer 3000 are both priced at \$990.

For further information, please contact:

For more information, contact:

**Teledyne LeCroy**  
www.teledynelecroy.com

*If you have a new product that you would like us to run in our New Products section, please email a short description (300-500 words) and a photo of your product to:*  
**newproducts@nutsvolts.com**

# CLASSIFIEDS

## SURPLUS

### SURPLUS ELECTRONIC PARTS & ACCESSORIES



Over  
20,000  
Items  
in  
Stock

Belts  
Cables  
Connectors  
Fans  
Hardware  
LEDs  
Motors  
Potentiometers  
Relays  
Semiconductors  
Service Manuals  
Speakers  
Switches  
Test Equipment  
Tools  
VCR Parts

Surplus Material Components  
SMC ELECTRONICS  
www.smcelectronics.com

No Minimum Order  
Credit Cards and PAYPAL Accepted.  
Flat \$4.95 per order USA Shipping.

## HARDWARE WANTED

### DEC EQUIPMENT WANTED!!!

Digital Equipment Corp.  
and compatibles.  
Buy - Sell - Trade

CALL KEYWAYS 937-847-2300  
or email buyer@keyways.com

## LIGHTING



Automatic  
Strip Lighting  
www.ReactiveLighting.com

Want to keep  
learning more  
about electronics?  
Subscribe today!  
www.nutsvolts.com

## AUDIO/VIDEO

**PARTS**  
EXPRESS  
YOUR ELECTRONIC CONNECTION

OVER 18,000  
ELECTRONIC PARTS  
IN STOCK



Visit us online to order a  
FREE copy of our 2015 catalog  
and a special offer!

parts-express.com/nuts  
1-800-338-0531

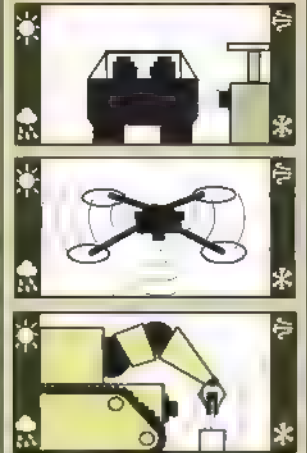
## WIRE/CABLE



**ANAHEIM WIRE, INC.**  
Master distributor of  
electrical and electronic  
wire and cable since  
1973. Items available from  
stock: Hook up wire, Shrink  
tubing, Cable ties, Connectors,  
etc. Wire cut & strip to specs,  
twisting, stripping. If interested,  
please call **1-800-626-  
7540**, FAX: 714-563-8309  
Visa/MC/Amex. See us on  
the Internet: www.anaheim  
wire.com or email: info@  
anaheimwire.com.

## ROBOTICS

### Need sensors?



www.maxbotix.com

## SERVICES

### Sandpiper Electronics Services LLC

"design of custom electronics instrumentation"

Circuit design & drawings, PCB layout, fabrication & population  
Prototype development for research, industry and events  
39 years experience in electronics development for research &  
flight applications, laser systems & laboratory research

FREE no obligation consultations  
www.sandtronic.com

Like to build robots?  
Then, you need a  
subscription to  
**SERVO Magazine!**

www.servomagazine.com

Did You Know Preferred Subscribers get access to  
all the digital back issues of Nuts & Volts for free?

Call for details: **1-877-525-2539**

## >>> QUESTIONS

### Variable Speed Motor

I'm looking to purchase a belt grinder for my workshop to smooth the edge of circuit boards, aluminum cutouts, and the output of my 3D printer. My question relates to the variable voltage controllers and three-phase motors that are available as optional equipment. Given almost double the price of the grinder, can I simply use a surplus variac on a single-phase motor to get a variable speed grinder?

#10151

**Dennis Brown**  
San Antonio, TX

### Confused

I seem to go through fuses quickly on my bench power supply. Would it be okay to try a higher than normal value fast-blow fuse or go with a slow-blow fuse of the original value?

#10152

**Roy Myrick**  
North Miami, FL

### Gold vs. Tin

I need to buy IC sockets in bulk for an upcoming project, and I'm debating whether the added cost is worth it to upgrade from tin to gold contacts. Am I paying for longevity or simply slightly lower contact resistance when I spend double or triple for a gold IC socket?

#10153

**Ted Walden**  
Fairbanks, AK

### Remote IR Sensor

My family runs a restaurant and does a lot of deep frying. I was thinking of setting up an IR remote

temperature monitor for the oil vs. an immersion sensor that would have to be cleaned. Is there a downside to using a remote sensor? Can you recommend a source?

**John Ladner**  
#10154 Minneapolis, MN

## >>> ANSWERS

### [#6152 - June 2015] SW Radio Info Needed

*I am interested in building the shortwave radio shown in the schematic from Michael Williams Tech Forum question #2153 on page 78 of the February 2015 issue. However, I need more information, specifically the dimensions of L1 – length, diameter, etc., and the frequency range of the receiver.*

*I'm also seeking information on the coil data, size, number of turns, etc., for the shortwave and broadcast band coils for the Allied Space Spanner regenerative receiver, as well as modifications (plug-in coils VFO) that would increase the frequency coverage above and below the stock range of 6 to 12 MHz on the shortwave section of the receiver*

First, I cannot help you with the information you are looking for on the Allied Space Spanner receiver. However, the other part of your question is about calculating the information for coil L1. The formula for calculating inductance needed for a resonate frequency as the value of the capacitor is shown.

To calculate the inductance of L1 for the standard broadcast band

(535 kHz to 1,700 kHz),  $L = \frac{25330}{f^2}$  where  $L$  = the inductance in  $\mu\text{H}$ ,  $f$  = the frequency in MHz, and capacitance is in pF. Since the capacitance is given as 10 to 365 pF and we know the minimum frequency is 535 kHz (or .535 MHz),  $L = 25330 / (.535 \times .535 \times 365)$ ;  $L = 242 \mu\text{H}$ . In the above formula, we replace the C and L can be interchanged. So, if we know the value of the capacitance, we can calculate the value of the inductance and vice versa.

With the calculated inductance (242  $\mu\text{H}$ ) and the maximum frequency of the broadcast band (1,700 kHz or 1.7 MHz),  $C = \frac{25330}{(1.7 \times 1.7 \times 242)}$ ;  $C = 32 \text{ pF}$ . So, with a coil wound very close to 242  $\mu\text{H}$ , the receiver would cover the standard broadcast band, and a little higher.

Let's now calculate the number of turns of #30 and diameter of the coil needed to get 242  $\mu\text{H}$ .  $N = \sqrt{\frac{L(9r+10l)}{r}}$  is the formula for calculating the number of turns on a coil, but we have to know the values of radius, and the length of the coil. With a coil, the larger the diameter and the closer the turns are together increases the inductance;  $r$  = the mean radius,  $l$  = the length of the coil.

I recently did a project very similar to this, so I have first-hand knowledge of the approximate dimensions of the coil. Looking in a wire table, I find that #30 wire will make approximately 90 turns for a linear inch. I then choose the length of the coil to be one inch long and the diameter to be two inches, so  $r = 1$ . After plugging the values into the formula, 67 turns is the result. So, close wind 67 turns of #30 wire on a

All questions AND answers are submitted by Nuts & Volts readers and are intended to promote the exchange of ideas and provide assistance for solving technical problems. All submissions are subject to editing and will be published on a space available basis if deemed suitable by the publisher. Answers are submitted by readers and

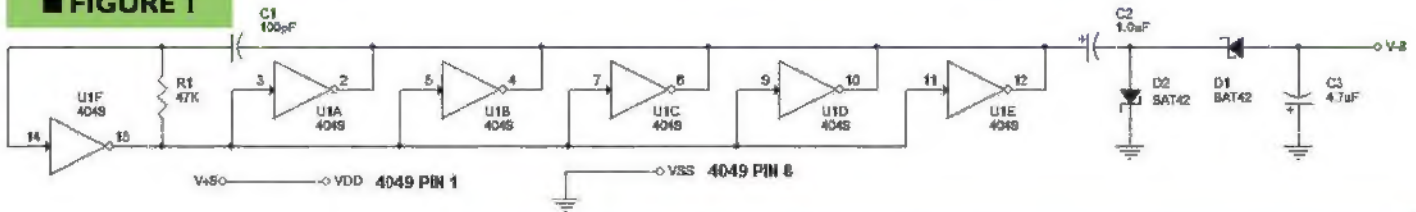
**NO GUARANTEES WHATSOEVER** are made by the publisher. The implementation of any answer printed in this column may require varying degrees of technical experience and should only be attempted by qualified individuals.

*Always use common sense and good judgment!*



Send all questions and answers by email to [forum@nutsvolts.com](mailto:forum@nutsvolts.com) or via the online form at [www.nutsvolts.com/tech-forum](http://www.nutsvolts.com/tech-forum)

■ FIGURE 1



two inch diameter form, then loosen the turns slightly, and spread them out so the length of the coil is one inch. With this information, you can calculate the other resonate circuits.

Ned Stevens  
Saint George, UT

**[#8152 - August 2015]  
Voltage Mod**

I am trying to build a small micro ampere meter project I found on the Internet (Figure 1). The design calls for a +9V and -9V supply. Is it possible to modify this circuit to use a single 9V battery instead of two?

That's a very old circuit and it has at least one glaring flaw: The value of R3 is way too high. I suspect it should be 3.3K instead. Also, D3 and D4 seem unnecessary because there can never be more than about 2 mA of meter current. R1 protects D1 and D2, but microamp circuit levels wouldn't threaten them. The input bias current of a 741 op-amp through a 10K resistor produces up to 5 mV offset that's not temperature-compensated. It's nullable, of course, but the null can drift.

While it's pretty easy to make a DC voltage inverter with circuits like the one in Figure 1, I believe there's a better solution. Single-supply op-amps are available with offset null pins, completely eliminating the negative power supply! See Figure 2 for my (untested) circuit suggestion. TLE2021 chips are available in plastic DIP packaging from the usual suspects, such as Digi-Key and Mouser.

The current sampling resistor should be selected for full-scale

# ALL ELECTRONICS

[www.allelectronics.com](http://www.allelectronics.com)

Order Toll Free 1-800-826-5432

**BATTERY HOLDER FOR 2 AA CELLS**

2.25" x 1.25" x 0.61" h gh. Unbreakable back past c. 6" w re eads.

CAT# BH-32 **85¢ each**

10 for 75¢ each • 100 for 60¢ each



**IEC POWER RECEPTACLE**

3 prong IEC power receptac e. Mount ng ears w th ho es on 1.575" centers. 0.25" qc / so der connectors. Rated 15A 250Vac. UL, CSA.

CAT# ACS-48

**\$1.25 each**

10 for \$1.10 each  
100 for 90¢ each



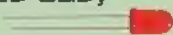
**12V RED DIFFUSED LED, 5MM (T1 3/4)**

Br ght 5mm round red LED w th bu t n res stors for 12Vdc, 12mA operat on. No externa res stor requ red. Works we on 4 12 Vdc. 35° v ew ng ang e.

CAT# LED-12R

**50¢ each**

10 for 45¢ each  
100 for 40¢ each



**20 AMP BARRIER STRIP**

4 pos t on dua row str p. Accepts w re from 22 to 14 AWG. 0.375" centers. 0.85" w de x 0.52" h gh x 2.16" ong. Mount ng ho es on 8mm centers.

CAT# TS-204

**\$1.35 each**

25 for \$1.17 each  
100 for 97¢ each



**SELF-FUSING RUBBER TAPE**

Stretch and wrap for a mo sture t ght, nsu at ng sea on e ect ca connect ons. H gh res stance to sa t water, steam and o . Protects from mo sture, most chem ca s, and nsu ates to 490V per m . 1" x 16.4' ro .

CAT# SFT-5

**\$4.50 each**



**12 VDC 5 AMP POWER SUPPLY**

Input: 100 240 Vac, 50/60 Hz, 1.6A. Output: 12 Vdc 5A. 5' cord w th ferr te sp t bead and 2.5mm coax power p ug. Center+, cULus.

CAT# PS-1262

**\$18.75 each**



**1" SPEAKER IN ENCLOSURE**

Great sound ng 8 Ohm speaker. Sea ed 1.25" square x 0.90" deep enc osure. My ar cone w th rubber surround. 2.5" eads w th 2 p n (2mm) socket connector.

CAT# SK-61

**\$2.00 each**



**2-TONE 12VDC SIREN, WAVE 2**

Honeywe / Ademco WAVE2 106db dua tone s ren for home or commerc a a arm nsta at ons. Can be w red for warb e or steady tone. 4.32" x 3.28" x 2" w th ABS p ast c snap open h nged case. F ts s ng e gang e ect ca box or wa p ate. Corner or ce ng mount ng w th no add t ona brackets or hardware. 12 Vdc 500 Ma. UL.

CAT# ES-16

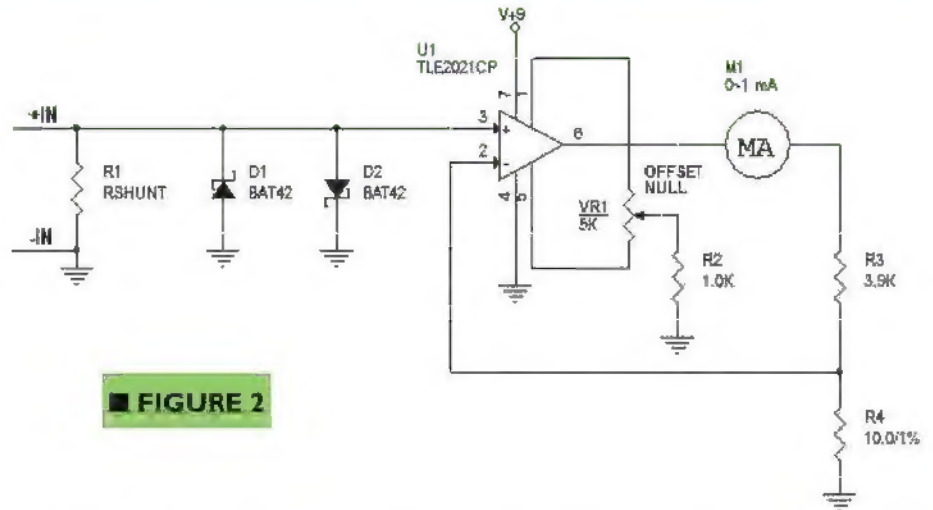
**\$4.25 each**



reading with a 10 mV drop, which is a 10X lower measurement burden relative to the original circuit.

If you add the input diode protection resistor back in, then you incur only 0.7 mV max uncompensated offset because of the TLE2021's enhanced performance. If you do this, then there's really no reason to keep those diodes — you can rely on the op-amp's input pin protection circuit.

**Mike Hardwick**  
Decade Engineering  
Turner, OR



■ FIGURE 2

■ LOOK FOR ■ SEARCH FOR ■ FIND  
Find your favorite advertisers here!

# ADvertiser INDEX

**AMATEUR RADIO**  
**ANDTV**  
National RF..... 21

**BUYING ELECTRONIC SURPLUS**  
All Electronics Corp. .... 81  
Earth Computer Technologies ..... 46

**CIRCUIT BOARDS**  
AP Circuits ..... 46  
Dimension Engineering ... 51  
ExpressPCB ..... 65  
Front Panel Express LLC ... 40  
Saelig Co. Inc. .... 2

**COMPONENTS**  
All Electronics Corp. .... 81

**COMPUTER Hardware**  
Earth Computer Technologies ..... 46

**Microcontrollers / I/O Boards**  
MikroElektronika ..... 3  
Technologic Systems ..... 59

**DATA ACQUISITION**  
Measurement Computing.47

**DESIGN/ENGINEERING/ REPAIR SERVICES**  
Cleveland Institute of Electronics ..... 52

ExpressPCB ..... 65  
Front Panel Express LLC.40  
National RF..... 21

**EDUCATION**  
Boxed Kit Amps ..... 21  
Cleveland Institute of Electronics ..... 52  
Command Productions ... 15  
littleBits ..... 67  
NKC Electronics ..... 21  
Parallax ..... 52  
Poscope ..... 09  
ServoCity ..... 83

**ENCLOSURES**  
Front Panel Express LLC 40

**HALLOWEEN**  
Skeletons & More ..... 21

**HI-FI AUDIO**  
Boxed Kit Amps ..... 21

**KITS & PLANS**  
Boxed Kit Amps ..... 21  
Earth Computer Technologies ..... 46  
littleBits ..... 67  
NKC Electronics ..... 21  
Parallax ..... 52  
Qkits ..... 21  
ServoCity ..... 83

**MISC./SURPLUS**  
All Electronics Corp. .... 81  
Front Panel Express LLC 40

**RF TRANSMITTERS/ RECEIVERS**  
Lemos International ..... 75  
National RF..... 21

**ROBOTICS**  
All Electronics Corp. .... 81  
Cleveland Institute of Electronics ..... 52  
littleBits ..... 67  
Parallax ..... 52  
ServoCity ..... 83

**TEST EQUIPMENT**  
Dimension Engineering ... 51  
Measurement Computing 47  
NKC Electronics ..... 21  
PicoTechnologyBack Cover  
Poscope ..... 09  
Saelig Co. Inc. .... 2  
Teledyne LeCroy ..... 35  
ZeroPlus Technology ..... 41

**TOOLS**  
MikroElektronika ..... 3  
PanaVise ..... 40  
Poscope ..... 09

All Electronics Corp. .... 81  
AP Circuits ..... 46  
Boxed Kit Amps ..... 21  
Cleveland Institute of Electronics ..... 52  
Command Productions ..... 15  
Dimension Engineering ..... 51  
Earth Computer Technologies 46  
ExpressPCB ..... 65  
Front Panel Express LLC ... 40  
Lemos International..... 75  
littleBits ..... 67  
Measurement Computing ... 47  
MikroElektronika ..... 3  
National RF..... 21  
NKC Electronics ..... 21  
PanaVise ..... 40  
Parallax ..... 52  
Pico Technology ....Back Cover  
Poscope ..... 09  
Qkits ..... 21  
Saelig Co., Inc. .... 2  
ServoCity ..... 83  
Skeletons & More ..... 21  
Technologic Systems ..... 59  
Teledyne LeCroy ..... 35  
ZeroPlus Technology ..... 41

**LINEAR MOTION  
MADE SIMPLE!**

# Actobotics™

**DREAM. DESIGN. BUILD. REPEAT.**

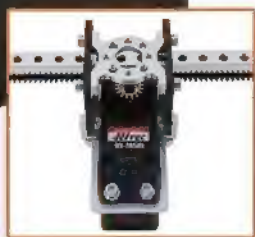
See the Gear Rack Kit in action!



#637170



## GEAR RACK KIT



JUST  
**\$89.99**  
(servo included)

The 785 Gear Rack Kit is a simple way to create linear motion using a rotational servo. The kit comes with the multi-rotation Hitec HS-785HB servo which allows for up to **9.6" of travel** when sending the proper PWM signal from a servo controller.



## CHANNEL SLIDER KIT

#637166



ONLY  
**\$169.99**  
(servo included)



See it in action!



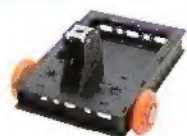
The XL timing belt and XL pulleys on this kit offer a reliable way to transfer the rotation of the servo into a **smooth linear motion** down the 24" channel backbone.



More

## NEW STUFF

added weekly!



\$89.99  
Gooseneck™



\$4.99  
Shaft Couplers



\$1.99  
80/20® Mounts



\$4.99  
\$3.99  
Linear Bearings



\$6.49  
\$4.49  
V-Rollers



\$169.99  
RoboClaw 2x45A



\$5.99  
Press-Fit Wheels



\$8.99  
Pinion Pulleys



\$29.99  
Dual Servo Driver



\$49.99  
Planetary Motors w/ Encoders



\$199.99  
The Warden™



\$79.99  
Gear Drive Pan



\$0.69 - \$0.99  
Power Input Boards



\$69.99  
The Bogie™



\$179.99  
177 pieces!  
Actobotics® Assortment Pack



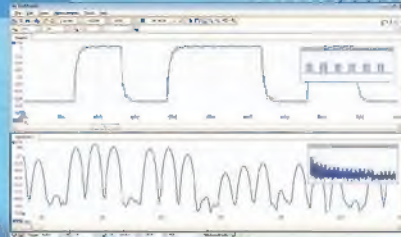
# SERVOCITY.com

620.221.0123  
sales@servocity.com

# PC OSCILLOSCOPES

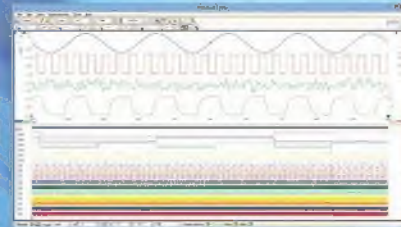


Low cost



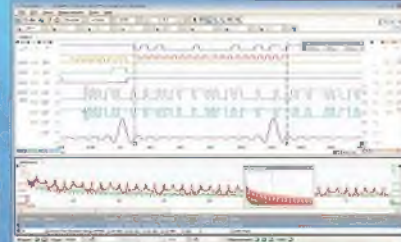
- 100 MS to 1 G (depending on model)
- Resolution enhanced by software
- 8 bit to 12 bit ADC
- 100 MS to 1 G (depending on model)

MSO



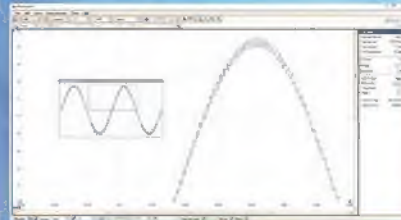
- 50 to 200 MHz bandwidth
- Resolution enhanced by software
- Memory for multiple channels
- ADC
- 50 to 200 MHz bandwidth

Eight channels



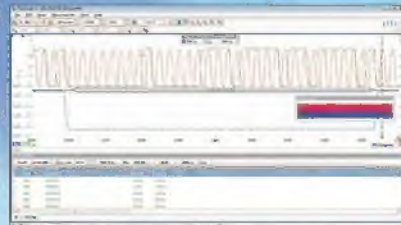
- 2 bit resolution (16 bit enhanced)
- Buffer
- 8 channels
- 2 bit resolution (16 bit enhanced)

Flexible resolution



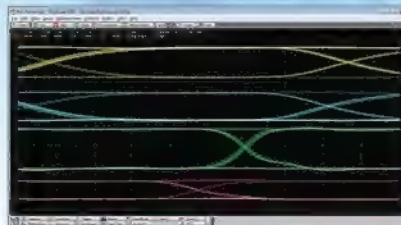
- 8, 12, 14, 16 bit resolution
- 8 to 51 MS buffer memory
- ADC
- 8, 12, 14, 16 bit resolution

2 GS memory



- 5 GS memory
- Utilize Gigabyte memory
- ADC
- 5 GS memory

20 GHz sampling



- 1 ps rise time
- 60 dB dynamic range
- ADC with 100 MS
- Precise 100 MS
- 1 ps rise time

Full software included as standard with serial bus decoding and analysis (CAN, LIN, RS232, I2C, I2S, SPI, FlexRay), segmentation, mask testing, spectrum analysis, and software development kit (SDK) all as standard, with free software updates. Five years warranty real time oscilloscopes, 2 years warranty sampling oscilloscopes.

1-800-591-2796  
www.picotech.com/pco544